



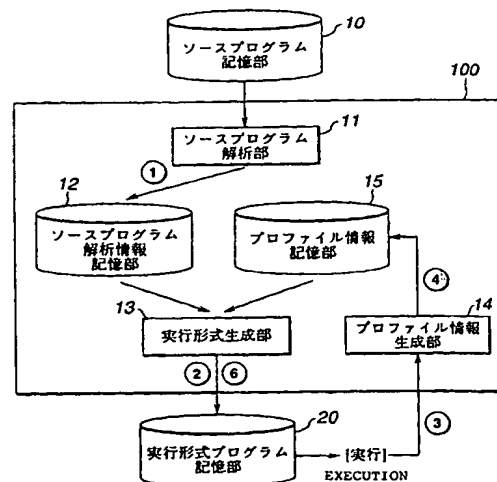
(51) 国際特許分類6 G06F 9/45	A1	(11) 国際公開番号 WO00/03322 (43) 国際公開日 2000年1月20日(20.01.00)
(21) 国際出願番号 PCT/JP99/03710 (22) 国際出願日 1999年7月8日(08.07.99) (30) 優先権データ 特願平10/195586 1998年7月10日(10.07.98) JP 特願平11/188661 1999年7月2日(02.07.99) JP (71) 出願人 (米国を除くすべての指定国について) ソニー株式会社(SONY CORPORATION)[JP/JP] 〒141-0001 東京都品川区北品川6丁目7番35号 Tokyo, (JP) (72) 発明者; および (75) 発明者/出願人 (米国についてのみ) 福澤祐二(FUKUZAWA, Yuji)[JP/JP] 岡田徹也(OKADA, Tetsuya)[JP/JP] 〒141-0001 東京都品川区北品川6丁目7番35号 ソニー株式会社内 Tokyo, (JP) (74) 代理人 小池 晃, 外(KOIKE, Akira et al.) 〒105-0001 東京都港区虎ノ門二丁目6番4号 第11森ビル Tokyo, (JP)		(81) 指定国 CA, US 添付公開書類 国際調査報告書

(54)Title: METHOD AND APPARATUS FOR COMPILING

(54)発明の名称 コンパイル処理装置及びコンパイル処理方法

(57) Abstract

A source program analyzer (11) performs various analyses of a source program, generates analysis information on the source program, and stores it in a source program analysis information storage (12). Based on the analysis information in the source program analysis information storage (12), an executable-program generator (13) generates an executable program for generating profile information. A profile information generator (14) generates profile information based on the information obtained by the executing of the program. The executable-program generator (13) generates an optimized executable program based on the analysis information on the source program stored in the source program analysis information storage (12) and the profile information generated by the profile information generator (14).



- 10 ... SOURCE PROGRAM STORAGE
- 11 ... SOURCE PROGRAM ANALYZER
- 12 ... SOURCE PROGRAM ANALYSIS INFORMATION STORAGE
- 13 ... EXECUTABLE-PROGRAM GENERATOR
- 14 ... PROFILE FILE INFORMATION GENERATOR
- 15 ... PROFILE FILE INFORMATION GENERATOR
- 20 ... EXECUTABLE-PROGRAM STORAGE

(57)要約

ソースプログラム解析部 1.1 は、ソースプログラムに対する種々の解析処理を行って、ソースプログラムの解析情報を生成し、これをソースプログラム解析情報記憶部 1.2 に記憶する。ソースプログラム解析情報記憶部 1.2 の解析情報に基づいて、実行形式生成部 1.3 によりプロファイル情報生成用実行形式プログラムを生成し、これを実行することにより得られる情報から、プロファイル情報生成部 1.4 によりプロファイル情報を生成する。実行形式生成部 1.3 は、ソースプログラム解析情報記憶部 1.2 に記憶されているソースプログラムの解析情報と、プロファイル情報生成部 1.4 により生成されたプロファイル情報とに基づいて、最適化した実行形式プログラムを生成する。

PCTに基づいて公開される国際出願のパンフレット第一頁に掲載されたPCT加盟国を同定するために使用されるコード(参考情報)

AE アラブ首長国連邦	DM ドミニカ	KZ カザフスタン	RU ロシア
AL アルバニア	EE エストニア	LC セントルシア	SD スーダン
AM アルメニア	ES スペイン	LI リヒテンシュタイン	SE スウェーデン
AT オーストリア	FI フィンランド	LK スリ・ランカ	SG シンガポール
AU オーストラリア	FR フランス	LR リベリア	SI スロヴェニア
AZ アゼルバイジャン	GA ガボン	LS レソト	SK スロヴァキア
BA ボスニア・ヘルツェゴビナ	GB 英国	LT リトアニア	SL シエラ・レオネ
BB バルバドス	GD グレナダ	LU ルクセンブルグ	SN セネガル
BE ベルギー	GE グルジア	LV ラトヴィア	SZ スワジランド
BF ブルキナ・ファソ	GH ガーナ	MA モロッコ	TD チャード
BG ブルガリア	GM ガンビア	MC モナコ	TG トーゴ
BJ ベナン	GN ギニア	MD モルドヴァ	TJ タジキスタン
BR ブラジル	GW ギニア・ビサウ	MG マダガスカル	TZ タンザニア
BY ベラルーシ	GR ギリシャ	MK マケドニア旧ユーゴスラヴィア	TM トルクメニスタン
CA カナダ	HR クロアチア		TR トルクメニスタン
CF 中央アフリカ	HU ハンガリー	ML マリ	TT トリニダード・トバゴ
CG コンゴ	ID インドネシア	MN モンゴル	UA ウクライナ
CH スイス	IE アイルランド	MR モリタニア	UG ウガンダ
CI コートジボアール	IL イスラエル	MW マラウイ	US 米国
CM カメルーン	IN インド	MX メキシコ	UZ ウズベキスタン
CN 中国	IS アイスランド	NE ニジェール	VN ヴイエトナム
CR コスタ・リカ	IT イタリア	NL オランダ	YC ユーゴスラビア
CU キューバ	JP 日本	NO ノルウェー	ZA 南アフリカ共和国
CY キプロス	KE ケニア	NZ ニュージーランド	ZW ジンバブエ
CZ チェコ	KG キルギスタン	PL ポーランド	
DE ドイツ	KP 北朝鮮	PT ポルトガル	
DK デンマーク	KR 韓国	RO ルーマニア	

明 細 書

コンパイル処理装置及びコンパイル処理方法

技 術 分 野

この発明は、例えば、逐次方式、並列方式、あるいは、V L I W (V e r y L o n g I n s t r u c t i o n W o r d) 方式などの計算機上でプログラミング言語で記述されたプログラムから実行形式のプログラムを生成するコンパイル処理装置に関する。

背 景 技 術

プログラミング言語で記述されたソースプログラムから、目的の電子計算機において実行可能な形式のプログラム（以下、この明細書においては実行形式プログラムという）を生成したり、最適化された実行形式プログラムを生成する場合に、コンパイル処理装置が用いられる。

例えば、図1に示すように、プログラマによって作成され、ソースプログラム記憶部1に記憶されたプログラミング言語で記載されたソースプログラムは、コンパイル処理装置2に入力される。コンパイル処理装置2は、ソースプログラムの字句解析、構文解析、意

味解析などを行って、ソースプログラムから目的の電子計算機で実行可能な実行形式プログラムを生成し、これを実行形式プログラム記憶部 3 に記憶する。

この実行形式プログラム記憶部 3 に記憶された実行形式プログラムを実行することにより得られる情報から、その実行形式プログラムのプロファイル情報が得られ、このプロファイル情報がプロファイル情報記憶部 4 に記憶される。プロファイル情報は、例えば、実行されたプログラムのどの部分が何回実行され、その実行時間がどれくらいであったかなど、プログラムを実行することにより得られるプログラムの動作に関する様々な情報を含んでいる。

そして、図 1 に示すように、コンパイル処理装置 2 に、ソースプログラム記憶部 1 に記憶されているプログラミング言語で記載されたソースプログラムと、プロファイル情報記憶部 4 に記憶されたそのプログラムに対するプロファイル情報とを入力して、再度コンパイル処理を実行することにより、プログラムの最適化を行って、最適化され、高速化された実行形式プログラムが生成され、これが実行形式プログラム記憶部 3 に記憶される。

ところで、図 1 を用いて前述したように、最適化された実行形式プログラムを生成する場合、まず、ソースプログラムをコンパイル処理装置に入力して、目的とする電子計算機の実行形式プログラムを生成し、この実行形式プログラムを実行することにより得られる情報から、プロファイル情報を得る。

そして、コンパイル処理装置に入力したソースプログラムと、プロファイル情報とをコンパイル処理装置に入力し、再度コンパイル処理を行うことにより、最適化された実行形式プログラムを生成さ

れる。

このため、プロファイル情報を得るために実行形式プログラムを生成するための最初のコンパイル処理と最適化されたプログラムを得るための再度のコンパイル処理において、ソースプログラムの字句解析、構文解析、意味解析などの解析処理が重複して行われるなどの無駄が生じ、最適化された実行形式プログラムを生成するまでに時間がかかってしまうという問題がある。

発 明 の 開 示

上述の如き従来の問題点に鑑み、本発明は、処理時間を短縮し、迅速に実行形式プログラムを生成することができるコンパイル処理装置及びコンパイル処理方法を提供することを目的とする。

本発明に係るコンパイル処理装置は、ソースプログラムから、そのソースプログラムの解析情報を生成する解析情報生成部と、前記解析情報に基づいて、第1の実行形式プログラムを生成する第1の実行形式プログラム生成部と、前記第1の実行形式プログラムに基づいて、プロファイル情報を生成するプロファイル情報生成部と、前記前記解析情報と、前記プロファイル情報とに基づいて、第2の実行形式プログラムを生成する第2の実行形式プログラム生成部とを備えることを特徴とする。

本発明に係るコンパイル処理方法は、ソースプログラムから、そのソースプログラムの解析情報を生成する解析情報生成工程と、前記解析情報に基づいて、第1の実行形式プログラムを生成する第1

の実行形式プログラム生成工程と、前記第 1 の実行形式プログラムに基づいて、プロファイル情報を生成するプロファイル情報生成工程と、前記解析情報と前記プロファイル情報とに基づいて、第 2 の実行形式プログラムを生成する第 2 の実行形式プログラム生成工程とを有することを特徴とする。

図面の簡単な説明

図 1 は、従来のコンパイル処理装置を用いて最適化された実行形式プログラムを生成する場合の手順を説明するための図である。

図 2 は、この発明によるコンパイル処理装置の一実施の形態を説明するための図である。

図 3 は、図 2 に示したコンパイル処理装置における実行形式生成部を説明するための図である。

図 4 は、ソースプログラムの一例を示す図である。

図 5 は、ソースプログラム解析部により生成される解析情報としての中間コードの一例を説明するための図である。

図 6 は、ソースプログラム解析部により生成される解析情報としてのアセンブリコードの一例を説明するための図である。

図 7 は、プロファイル情報生成部により生成されるプロファイル情報の一例を説明するための図である。

図 8 は、実行形式生成部の確率情報生成部において生成される確率情報の一例を説明するための図である。

図 9 は、プロファイル情報から生成される確率情報が用いられて最適化されたプログラムの例を説明するための図である。

図 1 0 は、コンパイル処理装置の実行形式生成部の他の例を説明するための図である。

図 1 1 は、この発明によるコンパイル処理装置の一実施の形態の他の例を説明するための図である。

図 1 2 は、この発明によるコンパイル処理装置の一実施の形態の他の例を説明するための図である。

図 1 3 は、図 1 2 に示したコンパイル処理装置の指示情報受付部と実行形式生成部とを説明するための図である。

図 1 4 は、コンパイルを補助するためのプログラムとしてのグラフィカルユーザインターフェースの一例を説明するための図である。

図 1 5 は、操作者からのコンパイル指示情報をも考慮して最適化されたプログラムの一例を説明するための図である。

図 1 6 は、この発明によるコンパイル処理装置の一実施の形態の他の例を説明するための図である。

発明を実施するための最良の形態

以下、本発明の実施の形態について図面を参照しながら詳細に説明する。

以下、図を参照しながらこの発明によるコンパイル処理装置及びコンパイル処理方法の一実施の形態について説明する。

[第 1 の実施の形態]

図 2 は、この第 1 の実施の形態のコンパイル装置 1 を説明するた

めの図である。図2に示すように、この第1の実施の形態のコンパイル処理装置100は、ソースプログラム記憶部10に記憶されている高水準プログラミング言語で記述されたソースプログラムから、目的の電子計算機において実行可能な実行形式プログラムを生成し、これを実行形式プログラム記憶部20に記憶するものである。

さらに、コンパイル処理装置100は、実行形式プログラム記憶部20に記憶された実行形式プログラムを実行することにより得られる情報からプロファイル情報を生成し、このプロファイル情報を用いて、実行形式プログラムを生成する処理を行うことにより、最適化された実行形式プログラムを生成することができるものである。

この第1の実施の形態のコンパイル処理装置100について説明する。この第1の実施の形態のコンパイル処理装置100は、図2に示すように、ソースプログラム解析部11、ソースプログラム解析情報記憶部12、実行形式生成部13、プロファイル情報生成部14、プロファイル情報記憶部15を備えている。

ソースプログラム記憶部10に記憶されている高水準プログラミング言語で記述されたソースプログラムは、コンパイル処理装置100のソースプログラム解析部11に供給される。ソースプログラム解析部11は、解析情報生成部としての機能を有しており、ソースプログラムの文字解析、構文解析、意味解析などの解析処理を行って、ソースプログラムからそのソースプログラムの解析情報を生成し、これをソースプログラム解析情報記憶部12に記憶する。

このソースプログラム解析情報記憶部12に記憶されるソースプログラムの解析情報は、ソースプログラムを解析することにより生成された、例えば、中間コード、アセンブリコード、機械語などの

生成情報を含むものであり、実行形式プログラムを生成するために必要な情報を含むものである。

そして、実行形式生成部 13 は、ソースプログラム解析情報記憶部 12 に記憶された解析情報から、目的とする電子計算機において実行される実行形式プログラムを生成し、これを実行形式プログラム記憶部 20 に記憶する。これによって、目的とする電子計算機において実行可能な実行形式プログラムが、実行形式プログラム記憶部 20 に得られる。

しかし、この段階の実行形式プログラムは、ソースプログラムからそのまま生成されたものであり、最適化されていない。このため、この実行形式プログラムを実行しても、無駄な処理が行われ、高速な処理が行えない場合もある。

そこで、コンパイル処理装置 100 のプロファイル情報生成部 14 は、実行形式プログラム記憶部 20 に記憶されている目的とする実行形式プログラムを実行することにより得られる情報から、プロファイル情報を生成し、これをプロファイル情報記憶部 15 に記憶する。

そして、コンパイル処理装置 100 において、ソースプログラム解析情報記憶部 12 に記憶されている最初のコンパイル処理によって生成されたソースプログラムの解析情報と、プロファイル情報記憶部 15 に記憶されているプロファイル情報とを用いて、実行形式生成部 13 により、再度の実行形式プログラムの生成処理を行う。

この再度の実行形式プログラムの生成処理は、最適化された実行形式プログラムを生成するための処理である。この処理は、図 2 に示すように、ソースプログラム解析情報記憶部 12 に記憶された解

析情報から生成されるプログラムに対して、プロファイル情報記憶部 15 のプロファイル情報を用いて最適化を行い、無駄な動作を行うことなく高速化された実行形式プログラムを生成し、これを実行形式プログラム記憶部 20 に記憶するものである。

図 3 は、この実施の形態のコンパイル処理装置 100 の実行形式生成部 13 を説明するための図である。この実施の形態の実行形式生成部 13 は、図 3 に示すように、プログラム構築部 131 と、最適化した実行形式プログラムを生成するために、プロファイル情報からプログラム実行についての確率情報を生成する確率情報生成部 132 を備えている。

前述したように、最初（1回目）のコンパイル処理においては、目的とするプログラムについてのプロファイル情報がまだプロファイル情報記憶部 15 に生成されていない。このため、実行形式生成部 13 は、目的とするソースプログラムについての最初のコンパイル処理においては、ソースプログラム解析情報記憶部 12 に記憶された中間コードやアセンブリコードなどの解析情報のみから、例えば、目的とする電子計算機において実行可能な実行形式プログラムを生成し、これを実行形式プログラム記憶部 20 に記憶する。

そして、前述したように、コンパイル処理装置 100 のプロファイル情報生成部 14 は、実行形式プログラム記憶部 20 に記憶されている最適化前の実行形式プログラムを実行することにより得られる情報から、プロファイル情報を生成し、これをプロファイル情報記憶部 15 に記憶する。

プロファイル情報記憶部 15 に記憶されるプロファイル情報は、実行形式プログラムの実行を開始してからの経過時間と、各経過時

間における詳細情報とからなるものである。そして、前述もしたように、プロファイル情報記憶部 15 に記憶されたプロファイル情報を用いて再度の実行形式プログラムの生成処理が行われることにより、最適化された実行形式プログラムが生成される。

この場合には、ソースプログラム解析情報記憶部 12 の中間コードやアセンブリコードなどの解析情報が、プログラム構築部 131 に供給されるとともに、プログラム実行の確率情報生成部 132 により、プロファイル情報記憶部 15 に記憶されているプロファイル情報を統計処理することにより生成されるプログラム実行の確率情報が、プログラム構築部 131 に供給される。

実行形式生成部 13 の確率情報生成部 132 により生成される確率情報は、後述もするが、例えば、プログラムを構成するブロックの実行回数、各命令の実行回数、分岐の回数、メモリやレジスタのアクセス回数などの各種の確率情報である。

そして、実行形式生成部 13 のプログラム構築部 131 は、ソースプログラム解析情報記憶部 12 からの解析情報から生成されるプログラムに対して、プロファイル情報記憶部 15 のプロファイル情報から生成された確率情報を用いて最適化を行い、無駄な動作を行うことなく高速化された実行形式プログラムを生成し、これを実行形式プログラム記憶部 20 に記憶する。

この場合、上述のように、ソースプログラム解析部 11 により各種の解析処理が行われて生成され、ソースプログラム解析情報記憶部 12 に記憶された解析情報が用いられて、最適化された実行形式プログラムが生成される。すなわち、最適化された実行形式プログラムを生成するために、従来のように、ソースプログラムの解析処

理を繰り返すことがないので、最適化された実行形式プログラムを生成するための処理にかかる時間を短縮し、迅速に最適化された実行形式プログラムを生成することができる。

そして、この第 1 の実施の形態のコンパイル処理装置 100 において行われる最適化された実行形式プログラムの生成手順は、以下に説明するように、第 1、第 2 の 2 つのプロセスからなっている。

すなわち、この第 1 の実施の形態のコンパイル処理装置 100 においては、まず、ソースプログラム記憶部 10 からソースプログラムを読み出し、そのソースプログラムをソースプログラム解析部 11 において字句解析、構文解析、意味解析などの解析処理を行うことにより、ソースプログラムの解析情報を生成し、これをソースプログラム解析情報記憶部 12 に記憶する（図 2 ①）。

そして、ソースプログラム解析情報記憶部 12 に記憶された解析情報から、プロファイル情報を得るために実行する実行形式プログラムを生成する（図 2 ②）。このプロファイル情報を得るためにソースプログラムから実行形式プログラムを生成するまでの処理が第 1 のプロセスである。

この第 1 のプロセスで生成された実行形式プログラムを実行することにより得られる情報に基づいて、プロファイル情報生成部 14 によりプロファイル情報を生成し（図 2 ③）、これをプロファイル情報記憶部 15 に記憶する（図 2 ④）。

そして、第 1 のプロセスにおいて生成され、ソースプログラム解析情報記憶部 12 に記憶されている解析情報と、プロファイル情報記憶部 15 に記憶されたプロファイル情報とに基づいて、実行形式生成部 13 において、再度の実行形式プログラムの生成を行うこと

により、プロファイル情報に基づいて最適化された実行形式プログラムが生成され、これが実行形式プログラム記憶部 20 に記憶される（図 2 ⑤）。このプロファイル情報の生成から、実行形式生成部 13 による再度の実行形式プログラムの生成までが第 2 のプロセスとなる。

そして、第 2 のプロセスにおいては、ソースプログラム解析情報記憶部 12 に記憶されている解析情報と、プロファイル情報とを用いて最適化した実行形式プログラムを生成するので、ソースプログラムの解析処理を行わなくても済む。

すなわち、第 1 のプロセスにおいて、ソースプログラム解析部 11 によりソースプログラムの解析処理を行った後においては、第 2 のプロセスにおいて、ソースプログラムの解析処理を繰り返し行う必要がないので、最適化された実行形式プログラムを生成するまでにかかる時間を短縮し、最適化された実行形式プログラムを迅速に生成することができる。

次に、この第 1 の実施の形態のコンパイル処理装置 100 において行われるコンパイル処理について、具体例を用いてより詳細に説明する。

図 4 は、ソースプログラム記憶部 10 に記憶されるソースプログラムの例を説明するための図である。この例のソースプログラムは、条件により変数 a 又は変数 b のどちらか一方を選択し、選択した変数の値を 2 乗したものを変数 d に格納するというものである。

そして、前述したように、ソースプログラム記憶部 10 に記憶されているソースプログラムを、ソースプログラム解析部 11 により解析することにより生成される、中間コード、アセンブリコード、

機械語などの解析情報（生成情報）が、ソースプログラム解析情報記憶部 12 に記憶される。この場合、中間コード、アセンブリコード、機械語などのうちいずれを生成するかは、例えば、ユーザにより選択することができるようにされる。

図 5、図 6 は、ソースプログラム記憶部 10 に記憶されている図 4 に示したソースプログラムがソースプログラム解析部 11 により解析されて、ソースプログラム解析情報記憶部 12 に生成される解析情報の例を説明するための図である。このうち、図 5 は、生成された解析情報が中間コードの場合の例を概念的に示し、図 6 は、生成された解析情報がアセンブリコードの場合の例を示している。

この例において、中間コード（図 5）の B L K 1 とアセンブリコード（図 6）の b l o c k 1 とは、変数 a と変数 b とを比較するブロックである。また、中間コード（図 5）の B L K 2 とアセンブリコード（図 6）の b l o c k 2 とは、変数 a が変数 b より大きい場合に、変数 c に変数 a を格納するブロックであり、中間コード（図 5）の B L K 3 とアセンブリコード（図 6）の b l o c k 3 とは、変数 b が変数 a 以上である場合に、変数 c に変数 b の値を格納するブロックである。そして、中間コード（図 5）の B L K 4 とアセンブリコード（図 6）の b l o c k 4 とは、選択された変数を 2 乗してその結果を変数 d に格納するブロックである。

なお、図 6 において、「l d」は、ロード命令、「s t」は、ストア命令、「c m p」は、比較命令、「b l e」は、分岐命令、「j m p」は、飛び越し命令、「m u l」は、掛け算命令である。したがって、図 6 に示すこの実施の形態のアセンブリコードの例の場合には、まず、b l o c k 1 の第 1 ステップにおいて、変数 a が

レジスタ r_1 に格納され、第 2 ステップにおいて、変数 b がレジスタ r_2 に格納される。

そして、block 1 の第 3 ステップにおいて、レジスタ r_1 とレジスタ r_2 との値が比較され、比較結果がレジスタ r_3 に格納される。そして、block 1 の第 4 ステップにおいて、レジスタ r_3 に格納された比較結果が、 $r_1 \leq r_2$ (変数 $a \leq$ 変数 b) であれば、block 3 への飛び越しが行われる。したがって、レジスタ r_3 に格納された比較結果が、 $r_1 > r_2$ (変数 $a >$ 変数 b) であれば、block 2 の処理が行われることになる。

block 2 は、上述のように block 1 におけるレジスタ r_1 の値とレジスタ r_2 の値との比較結果が、 $r_1 > r_2$ である場合に行われる処理である。この block 2 においては、第 1 ステップにおいて、変数 a がレジスタ r_4 に格納され、第 2 ステップにおいて、レジスタ r_4 に格納されている値が、変数 c に格納される。そして、block 2 の第 3 ステップにより、block 4 への飛び越しが行われる。

block 3 は、上述のように block 1 におけるレジスタ r_1 の値とレジスタ r_2 の値との比較結果が、 $r_1 \leq r_2$ である場合に行われる処理である。この block 3 においては、まず、第 1 ステップにおいて、変数 b が、レジスタ r_5 に格納され、第 2 ステップにおいて、レジスタ r_5 の値が、変数 c に格納される。そして、block 3 の第 3 ステップにより、block 4 への飛び越しが行われる。

そして、block 4 においては、第 1 ステップにおいて、変数 c がレジスタ r_6 に格納され、第 2 ステップにおいて、さらに変数

c が、レジスタ r 7 に格納される。そして、block 4 の第 3 ステップにおいて、レジスタ r 6 に格納されている値と、レジスタ r 7 に格納されている値との掛け算が行われ、第 4 ステップにおいて、第 3 ステップの掛け算の結果が、レジスタ r 8 に格納される。

このように、図 6 に示したこの実施の形態で用いるソースプログラムのアセンブリコードは、変数 a と変数 b の大小関係に応じて、変数 a の 2 乗、あるいは、変数 b の 2 乗を求め、レジスタ r 8 に格納するようにしたものである。

そして、実行形式生成部 1 3 は、前述のようにソースプログラム解析情報記憶部 1 2 に記憶された中間コードやアセンブリコードなどの解析情報から、目的とする電子機器において実行される形式の実行形式プログラムを生成し、これを実行形式プログラム記憶部 2 0 に記憶する。ここで、実行形式プログラム記憶部 2 0 に生成された実行形式プログラムは、最初（1 回目）のコンパイル処理により生成された実行形式プログラムであり、最適化前の実行形式プログラムである。

次に、コンパイル処理装置 1 0 0 のプロファイル情報生成部 1 4 は、実行形式プログラム記憶部 2 0 に記憶されている目的とする実行形式プログラムを実行することにより得られる情報から、プロファイル情報を生成し、これをプロファイル情報記憶部 1 5 に記憶する。

図 7 は、プロファイル情報生成部 1 4 により、プロファイル情報記憶部 1 5 に生成されるプロファイル情報の例を説明するための図である。この図 7 に示すプロファイル情報は、例えば、図 6 に示したアセンブリコードから生成された実行形式プログラムを実行する

ことにより得られる情報から生成されるものである。

図7に示すように、プロファイル情報記憶部15に記憶されるプロファイル情報は、実行形式プログラムの実行を開始してからの経過時間と、各経過時間における詳細情報とからなるものである。すなわち、プロファイル情報は、実行形式プログラム記憶部20に記憶されている実行形式プログラムを実行した場合に、時間の経過とともに、どの命令が実行され、どこで分岐し、また、どのメモリがアクセスされたか、あるいは、どのレジスタがアクセスされたかなど、プログラムの実行の状態を示すものである。

この図7に示したプロファイル情報の例は、以下のように、実行形式プログラムの実行を開始してからの各経過時間における詳細情報が示されている。つまり、この図7の例の場合には、経過時間10050において、プログラムカウンタ(PC)が変化し、図6に示したアセンブリコードのblock1が実行されて、変数aがレジスタに格納(ロード)されている。そして、経過時間10051において、変数bがレジスタに格納(ロード)され、経過時間10052において、比較命令(compare)が実行されている。

経過時間10053において、分岐命令(branch less or equal)が実行されて、図6に示したアセンブリコードのblock3を実行するようにされている。そして、経過時間10054において、プログラムカウンタ(PC)が変化し、図6に示したアセンブリコードのblock3が実行されて、変数bがレジスタに格納(ロード)されている。そして、経過時間10055において、レジスタの値が変数cに格納(ストア)され、経過時間10056において、図6に示したアセンブリコードのblo

c k 4 を実行するように飛び越し命令 (j u m p) が実行されている。

この後、経過時間 1 0 0 5 7 において、プログラムカウンタ (P C) が変化し、図 6 に示したアセンブリコードの b l o c k 4 が実行されて、変数 c がレジスタに格納 (ロード) され、経過時間 1 0 0 5 8 において、さらに変数 c の値が他のレジスタに格納 (ロード) されている。

そして、経過時間 1 0 0 5 9 において、掛け算命令 (m u l) が実行され、経過時間 1 0 0 6 0 において、経過時間 1 0 0 5 9 において実行された掛け算命令より求められた変数 d の値が、レジスタに格納 (ストア) されている。このように、実行形式プログラムを実行することにより得られた情報から、経過時間と、この経過時間に対応する実行形式プログラムの実行の状態の詳細情報が、プロファイル情報としてプロファイル情報記憶部 1 5 に記憶される。

そして、コンパイル処理装置 1 0 0 において、ソースプログラム解析情報記憶部 1 2 に記憶されている最初のコンパイル処理のソースプログラムの解析処理により、ソースプログラム解析情報記憶部 1 2 に生成されたソースプログラムの解析情報と、プロファイル情報記憶部 1 5 に記憶されているプロファイル情報とを用いて、実行形式生成部 1 3 により、再度の実行形式プログラムの生成処理を行う。

この場合には、図 3 に示したように、ソースプログラム解析情報記憶部 1 2 からの中間コードやアセンブリコードなどの解析情報と、確率情報生成部 1 3 2 により生成される確率情報とが、実行形式生成部 1 3 のプログラム構築部 1 3 1 に供給される。

図 8 は、プロファイル情報記憶部 1 5 に記憶されているプロファイル情報に基づいて、確率情報生成部 1 3 2 により生成される確率情報の例を説明するための図である。この図 8 に示す確率情報の例は、図 6 に示したアセンブリコードを実行形式プログラムに変換し、これを実行して得られる情報から形成されたプロファイル情報（図 7）に基づいて生成されたものである。

図 8 に示すように、確率情報生成部 1 3 2 は、前述にもしたように、プロファイル情報記憶部 1 5 に記憶されているプロファイル情報を統計処理することにより、各ブロックの実行回数や、各命令の実行回数、分岐の回数、メモリやレジスタのアクセス回数などの確率情報を生成する。

図 8 に示す確率情報の例の場合には、実行形式プログラムが実行されるごとに、b l o c k 1 と b l o c k 4 とは必ず実行されるが、b l o c k 2 が実行される確率（変数 a が変数 b より大きい確率）は、1 0 % であり、b l o c k 3 が実行される確率（変数 b が変数 a 以上である確率）は、9 0 % であることを示している。

そして、実行形式生成部 1 3 のプログラム構築部 1 3 1 は、ソースプログラム解析情報記憶部 1 2 の解析情報、この例の場合にはアセンブリコードと、図 8 に示した確率情報とに基づいて、最適化した実行形式プログラムを生成し、これを実行形式プログラム記憶部 2 0 に記憶する。

図 9 は、プロファイル情報から生成した確率情報を用いて最適化したプログラムの例を説明するための図である。この図 9 は、図 6 に示したアセンブリコードを確率情報を用いて最適化した実行形式プログラムをアセンブリコードで示したものである。

図8の確率情報に示したように、block 3が実行される確率は90%であり、block 2が実行される確率は10%であるので、block 1→block 3→block 4の経路が最適化されたものが図9に示す最適化されたプログラム（アセンブリコード）である。

この図9に示す最適化されたアセンブリコードと、図6に示した最適化される前のアセンブリコードとを比較すると分かるように、確率情報を用いて最適化を行うと、最適化後のアセンブリコードのblock 1の3行目には、新たにレジスタr2の値（変数b）をレジスタr5に格納するコードが追加され、block 3のコードと、block 4のコードが統合するようにされている。

すなわち、最適化が行われることにより、最適化前のblock 3の一部のコードが、最適化後のblock 1に移動され、block 3のコードと、block 4のコードが統合するようにされるとともに、レジスタの割り当ても変更されている。これにより、block 1において、レジスタr2の値（変数b）がレジスタr5に格納されているので、変数bが変数a以上であるときには、block 1に続くblock 3において、変数c（すなわち変数b）の2乗が即座に求められ変数dに格納することができるようにされる。すなわち、block 1とblock 3の2つのblockの処理で、変数bの2乗を求めることができるようにされる。

また、変数aが変数bより大きい場合には、図6に示した最適化前のアセンブリコードの場合と同様に、block 2において変数aがレジスタr4を介して変数cに格納され、block 4において、変数aが2乗されて変数dに格納されるというように、3つの

ブロックの処理で、変数 a の 2 乗を求めることができるようにされる。

このように、プロファイル情報から生成される確率情報を考慮した最適化が行われることにより、変数 b が変数 a 以上である確率が 90% と高いことが考慮されて、変数 b が、変数 a 以上である場合の処理を効率よく迅速に行うことができるように、最適化した実行形式のプログラムが生成される。

なお、図 9 においては示さなかったが、block 3 からジャンプする先の block 5 は、block 4 に続くブロックである。また、実行形式生成部 13 は、前述したように、命令の並べ換えや追加、レジスタの割当ての変更、分岐先の変更のほか、コードのコピー、不要なコードや命令の削除などを行って、最適化した実行形式のプログラムを生成することができるものである。

このように、この実施の形態のコンパイル処理装置 100 の実行形式生成部 13 は、最初（1回目）のコンパイル処理時においては、プロファイル情報生成用実行形式プログラム生成部として機能する。そして、プロファイル情報が生成された後においては、ソースプログラム解析情報記憶部 12 のソースプログラムと、プロファイル情報記憶部 15 のプロファイル情報から生成される確率情報とを用いて、最適化された実行形式プログラムを生成する最適化実行形式プログラム生成部として機能する。

そして、再度の実行形式プログラムの生成処理は、最適化された実行形式プログラムを生成するための処理である。この処理は、図 2 に示すように、ソースプログラム解析情報記憶部 12 に記憶された解析情報から生成されるプログラムに対して、プロファイル情報

記憶部 15 のプロファイル情報を用いて最適化を行い、無駄な動作を行うことなく高速化された実行形式プログラムを生成し、これを実行形式プログラム記憶部 20 に記憶するものである。

この場合、上述のように、ソースプログラム解析部 11 により各種の解析処理が行われて生成され、ソースプログラム解析情報記憶部 12 に記憶された解析情報が用いられて、最適化された実行形式プログラムが生成される。すなわち、最適化された実行形式プログラムを生成するために、従来のように、ソースプログラムの解析処理を繰り返すことがないので、最適化された実行形式プログラムを生成するための処理にかかる時間を短縮し、迅速に最適化された実行形式プログラムを生成することができる。

なお、実行形式プログラムには、直接実行形式と、間接実行形式の 2 つの実行形式があるが、この第 1 の実施の形態のコンパイル処理装置 100 の実行形式生成部 13 は、直接実行形式プログラム、間接実行形式プログラムのどちらも生成することができるものである。ここで、直接実行形式プログラムは、プログラムの実行が、プログラムを実行する機器に依存するものであり、間接実行形式プログラムは、プログラムの実行が、プログラムを実行する機器に依存することなく実行可能なものである。

そして、直接実行形式プログラムを生成するか、間接実行形式プログラムを生成するかは、例えば、解析情報記憶部 12 に記憶された解析情報に基づいて、自動的に切り換えることができるようにされる。例えば、解析情報記憶部 12 に記憶された解析情報が、目的とする計算機で実行する実行形式プログラムを生成するためのアセンブリコードやオブジェクトコードである場合には、直接実行形式

プログラムを生成し、また、解析情報記憶部 12 に記憶された解析情報が、中間コードである場合には、間接実行形式プログラムを生成するというように、自動的に切り換えを行うようにすることができる。

また、直接実行形式プログラムを生成するか、間接実行形式プログラムを生成するかは、例えば、コンパイル処理装置 100 に設けられた切り換えスイッチなどを操作することにより切り換えるようにすることもできる。

同様に、ソースプログラムから解析処理を行って実行形式プログラムを生成する第 1 のプロセスの実行形式プログラムの生成と、ソースプログラム解析情報記憶部 12 に記憶された解析情報を用いて実行形式プログラムを生成する第 2 のプロセスの実行形式の生成とは、例えば、コンパイル処理装置が、コンパイル処理の対象になっているプログラムについて、最初のコンパイル処理か、2 度目以降のコンパイルかを管理することにより、自動的に切り換えることができる。また、例えば、コンパイル処理装置 100 に設けられた切り換えスイッチなどを操作することにより、第 1 のプロセスの実行形式プログラムの生成と、第 2 のプロセスの実行形式の生成とを切り換えるようにすることもできる。

なお、前述の実施の形態においては、実行形式生成部 13 によりプロファイル情報生成用の実行形式プログラムと、最適化された実行形式プログラムとを生成するようにしたが、これに限るものではない。つまり、プロファイル情報を生成するための実行形式生成部（プロファイル情報生成用実行形式生成部）と、最適化された実行形式プログラムを生成する最適化実行形式生成部とを別々に設ける

ようにしてもよい。

ところで、前述にもしたように、ソースプログラム解析部 11 は、中間コード、アセンブリコード、機械語などを生成することができるものである。そして、解析情報記憶部 12 に記憶した解析情報が、例えば、中間コードなどの場合には、実行形式プログラムを構築しやすい形式のコードに変換して、実行形式プログラムを生成し、生成した実行形式プログラムをさらにオブジェクトコードに変換する必要が生じる。このような場合には、実行形式生成部 13 に、プリコードジェネレータと、ポストコードジェネレータとが必要になる。

図 10 は、ソースプログラム解析情報記憶部 12 に生成された解析情報が、例えば、中間コードなどの場合であって、この中間コードから実行形式プログラムを生成する場合の実行形式生成部 13 を説明するための図である。図 10 に示すように、この例の場合には、ソースプログラム解析情報記憶部 12 とプログラム構築部 131 との間にプリコードジェネレータ 133 を設けるとともに、プログラム構築部 131 の後段にポストコードジェネレータ 134 を設けたものである。

プリコードジェネレータ 133 は、解析情報記憶部 12 に記憶されている解析情報から、プログラム構築部 131 において実行形式プログラムを生成しやすい形式のコードを生成するものである。また、ポストコードジェネレータ 134 は、プログラム構築部 131 において、構築された実行形式プログラムから、オブジェクトコードの実行形式プログラムを生成するものである。

このため、解析情報記憶部 12 に記憶されている解析情報が、例えば、特定のプログラミング言語（例えば、J a v a）のバイトコ

ードなどである場合には、プリコードジェネレータ 1 3 3 により、実行形式プログラムの生成がしやすい形式の例えば中間コードに変換され、これがプログラム構築部 1 3 1 に供給される。

プログラム構築部 1 3 1 は、プリコードジェネレータ 1 3 1 から供給された例えば中間コードから、あるいは、プロファイル情報をも用いて、実行形式プログラムを構築する。この場合、構築された実行形式プログラムは、オブジェクトコードではない。そこで、ポストコードジェネレータ 1 3 4 は、プログラム構築部 1 3 1 において構築された例えば中間コードの実行形式プログラムから、オブジェクトコードの実行形式プログラムを生成し、これを実行形式プログラム記憶部 2 0 に記憶する。

このように、プリコードジェネレータ 1 3 3、ポストコードジェネレータ 1 3 4 を実行形式生成部 1 3 に備えることによって、ソースプログラム解析情報記憶部 1 2 に生成された解析情報が、中間コードや J a v a のバイトコードなどの場合であっても、オブジェクトコードの実行形式プログラムを生成することができる。

なお、プリコードジェネレータ 1 3 3 が、解析情報記憶部 1 2 に記憶された例えば中間コードなどの解析情報からオブジェクトコードを生成することができるものであれば、ポストコードジェネレータ 1 3 4 を用いなくてもすむようにすることができる。また、プログラム構築部 1 3 1 が、例えば、J a v a のバイトコードなどから、実行形式プログラムを構築することができるものであれば、プログラム構築部 1 3 1 において生成された実行形式プログラムから、ポストコードジェネレータ 1 3 4 により、オブジェクトコードの実行形式プログラムを生成するようになれば、プリコードジェネ

レータ 1 3 1 を用いなくてもすむようにすることができる。

このような場合には、プリコードジェネレータ 1 3 3 のみを設け、中間コードなどから実行形式プログラムを構築する場合には、常に、オブジェクトコードに変換した後に、実行形式プログラムを生成するようにしたり、また、ポストコードジェネレータ 1 3 4 のみを設け、中間コードなどから実行形式プログラムを構築する場合には、常に、中間コードなどからそのまま実行形式プログラムを生成し、中間コードなどの実行形式プログラムをポストコードジェネレータ 1 3 4 によりオブジェクトコードに変換するようにしてもよい。

また、プリコードジェネレータやポストコードジェネレータを備えた実行形式生成部と、これらを備えない実行形式生成部とを設けておくことにより、ソースプログラム解析情報記憶部 1 2 に生成される解析情報が、図 4 に示したような中間コードであっても、あるいは、図 6 に示したようなアセンブリコードであっても、それらから実行形式プログラムを生成することができる。

また、図 1 0 に示したように、実効形式生成部 1 3 にプリコードジェネレータ 1 3 3、ポストコードジェネレータ 1 3 4 を設けておき、アセンブリコードなどのオブジェクトコードが実行形式生成部 1 3 に供給されたときには、プリコードジェネレータ 1 3 3、ポストコードジェネレータ 1 3 4 をバイパスするようにしてもよい。

[第 2 の実施の形態]

ところで、プロファイル情報は、前述の第 1 の実施の形態においても説明したように、実行形式プログラムを実行することにより得られる情報から生成される。しかし、コンパイル処理装置により生成した実行形式プログラムを、コンパイル処理装置において簡単に

実行できない場合がある。

例えば、生成した実行形式プログラムが、コンパイル処理を行った装置以外の計算機用の実行形式プログラムであり、ハードウェアの違いにより、プログラムの命令セットが違ったり、あるいは、OS（オペレーティングシステム）が異なるなど実行形式プログラムの実行環境が異なる場合である。このような場合、当該コンパイル処理装置においては、生成した実行形式プログラムを実行することができず、簡単にはプロファイル情報を得ることができない。

また、目的とする実行形式プログラムが、例えば、VTRなどの電子機器の基板（ターゲットのボード）のメモリにインストールされて用いられるものである場合、例えば、テスト用の基板に実行形式プログラムをインストールして実行し、この実行により得られる情報をコンパイル処理装置に供給するようにしなければならない。

しかし、この場合には、テスト用の基板を予め用意するとともに、テスト用の基板と、コンパイル処理装置とを多数のコード（cord）によって接続するようにしなければならない。また、この場合には、コンパイル処理装置にテスト用の基板上で目的とする実行形式のプログラムを実行してプロファイル情報を得るためのシュミレータを搭載しておくことも考えられるが、シュミレータの開発に時間やコストがかかる。

そこで、この第2の実施の形態のコンパイル処理装置においては、プログラムの実行環境などに依存することなく実行が可能な、間接実行形式の中間コードプログラムを生成し、これを最適化した実行形式プログラムを生成する当該コンパイル処理装置において実行することにより、簡単かつ迅速にプロファイル情報を得て、最適化し

た目的とする実行形式プログラムを迅速に生成するようにしたものである。

図 1 1 は、間接実行形式の中間コードプログラムを生成することができるコンパイル処理装置 2 0 0 を説明するための図である。図 1 1 に示すように、この例のコンパイル処理装置 2 0 0 は、ソースプログラム解析部 1 1、ソースプログラム解析情報記憶部 1 2、実行形式生成部 1 3 A、1 3 B、1 3 C、プロファイル情報生成部 1 4、プロファイル情報記憶部 1 5、間接実行形式中間生成部 1 7 を備えている。

間接実行形式中間生成部 1 7 を備えるとともに、中間コード、アセンブリコード、機械語などもそれぞれから実行形式プログラムを生成する実行形式生成部 1 3 A、1 3 B、1 3 C を備える他は、図 1 を用いて前述した第 1 の実施の形態のコンパイル処理装置 1 0 0 と同様に構成されたものである。このため、図 1 1 に示すこの第 2 の実施の形態のコンパイル処理装置 2 0 0 おいて、第 1 の実施の形態のコンパイル処理装置 1 0 0 と同様に構成される部分には、図 2 に示したコンパイル処理装置 1 0 0 と同じ参照符号を付し、その説明については省略する。

なお、実行形式生成部 1 3 A、1 3 B、1 3 C のそれぞれは、異なる機器で実行される直接実行形式プログラムを生成するために設けられたものである。このように、種類の異なる実行形式プログラムを生成するために複数の実行形式生成部 1 3 A、1 3 B、1 3 C を設けるようにした場合であっても、プロファイル情報は、共通に用いることができる。

また、この実施の形態においては、例えば、実行形式生成部 1 3

Aは中間コードを使用し、実行形式生成部13Bは、アセンブリコードを使用し、実行形式生成部13Cは、その他のオブジェクトコードを使用するというように、生成する実行形式プログラムに応じて異なる解析情報を用いるようにされたものである。

しかし、実行形式生成部13A、13B、13Cのそれぞれは、基本的には、前述した第1の実施の形態において、図3あるいは図10を用いて前述した実行形式生成部13と同様に構成されたものであり、ソースプログラム解析情報記憶部12からの解析情報から実行形式プログラムを生成することができるとともに、プロファイル情報記憶部15のプロファイル情報から生成される確率情報をも用いることによって、最適化された実行形式プログラムを生成することができるものである。

そして、この図11に示すコンパイル処理装置200においては、間接実行形式中間生成部17により、ソースプログラム解析情報記憶部12に記憶されたソースプログラムの解析情報から、このコンパイル処理装置200において実行可能な間接実行形式の中間コードプログラムが生成され、これが中間プログラム記憶部30に記憶される。

この中間プログラム記憶部30に記憶された間接実行形式の中間コードプログラムを実行することにより得られる情報から、プロファイル情報生成部14によりプロファイル情報が生成されて、プロファイル情報記憶部15に記憶される。

そして、実行形式生成部13A、13B、13Cのいずれかにより、プロファイル情報記憶部15に記憶されたプロファイル情報と、ソースプログラム解析情報記憶部12に記憶されている解析情報と

から、目的とする計算機において実行可能とされた最適化された間接実行形式プログラムが生成される。この第2の実施の形態において、実行形式生成部13A、13B、13Cのいずれが用いられるかは、例えば、このコンパイル処理装置200の操作者によって指示される。

このように、最適化のために用いるプロファイル情報を生成するためにコンパイル処理装置により生成された実行形式プログラムが、当該コンパイル処理装置においてすぐに実行できないような場合でも、計算機に依存することなく実行が可能な間接実行形式の中間コードプログラムを生成して、実行することにより、目的とするプログラムに対するプロファイル情報を得て、目的の計算機において実行可能とされた間接実行形式の実行形式プログラムを迅速に生成することができる。

そして、この第2の実施の形態の場合には、それぞれの実行形式生成部13A、13B、13Cで生成された実行形式プログラムは、対応する実行形式プログラム20A、20B、20Cに記憶するようにされ、操作者の目的とする実行形式プログラムを生成することができる。

なお、前述の実施の形態においては、実行形式生成部13A、13B、13Cのうちいずれを用いるかは、操作者が選択するものとして説明したがこれに限るものではない。例えば、中間コード、アセンブリコード、機械語などの使用する解析情報を操作者が選択することにより、実行形式生成部が自動的に決まるようにすることもできる。つまり、使用する解析情報と、使用する実行形式生成部とを1対1に対応させるようにコンパイル処理装置200を構成する

こともできる。

また、使用する解析情報と、使用する実行形式生成部との両方を操作者が選択して、目的とする実行形式プログラムを生成するようにコンパイル処理装置 200 を構成するようにすることもできる。

また、複数の実行形式生成部を設けるのではなく、例えば、操作者からの指示に応じて、供給される解析情報が、中間コード、アセンブリコード、機械語、その他のオブジェクトコードなどであっても、それらに応じて実行形式プログラムを生成することができる多機能の実行形式生成部を設けるようにすることもできる。

このように、この第 2 の実施の形態のコンパイル処理装置 200 は、例えば、操作者からの指示に応じて、操作者が目的とする実行形式プログラムを生成することができるものである。

また、実行形式生成部に供給される解析情報が、中間コード、アセンブリコード、機械語などを使い分けることができるのと同様に、中間実行形式中間生成部 17 にも、中間コードを供給するか、アセンブリコードを供給するか、機械語を供給するかなどを切り換えることができるようにすることもできる。

したがって、中間実行形式中間生成部 17 に供給される解析情報と、実行形式生成部 13 A、13 B、13 C に供給される解析情報とが、異なる種類のコードとなる場合もあるし、同じ場合になることもある。要するに、種々の解析情報の使い分けをできるようにしておくことにより、より柔軟に各種の間接実行形式プログラム（中間コードプログラム）や直接実行形式プログラムを生成することができるようにされる。

また、間接実行形式中間生成部 17 により中間コードプログラム

を生成するのか、実行形式生成部 13A、13B、13Cのいずれかにより実行形式プログラムを生成するのかなどの切り換えは、例えば、コンパイル処理装置 100 に設けられた切り換えスイッチなどを操作することにより切り換えることができるようにされる。

〔第 3 の実施の形態〕

例えば、1つのコンパイル処理装置により直接実行形式と、間接実行形式とのいずれの実行形式プログラムをも生成可能にする場合や、プログラムの実行環境が異なる種々の計算機で実行される実行形式プログラムを生成しようとする場合には、直接実行形式と間接実行形式との別や、目的とする計算機のプログラムの実行環境などに関する情報などをコンパイル処理時にコンパイル処理装置に入力する必要が生じる。

そこで、この第 3 の実施の形態のコンパイル処理装置は、コンパイル処理装置の操作者からのコンパイル指示情報や、コンパイル処理を補助するためのプログラムから提供されるコンパイル指示情報を受け付けて、受け付けたコンパイル指示情報をも考慮して実行形式プログラムを生成することができるようにしたものである。

図 12 は、この第 3 の実施の形態のコンパイル処理装置 300 を説明するための図である。図 12 に示すように、この第 3 の実施の形態のコンパイル処理装置 300 は、ソースプログラム解析部 11、ソースプログラム解析情報記憶部 12、実行形式生成部 13、プロファイル情報生成部 14、プロファイル情報記憶部 15、指示情報受付部 18 を備えている。

指示情報受付部 18 以外の各部は、図 2 を用いて前述した第 1 の実施の形態のコンパイル処理装置 100 の各部と同様に構成された

ものである。このため、図 1 2 に示すこの第 3 の実施の形態のコンパイル処理装置 3 0 0 おいて、第 1 の実施の形態のコンパイル処理装置 1 0 0 と同様に構成される部分には、図 2 に示したコンパイル処理装置 1 0 0 と同じ参照符号を付し、その説明については省略する。

また、この例においても、ソースプログラム格納部 1 0 には、図 4 に示したソースプログラムが格納されており、このソースプログラムをソースプログラム解析部 1 1 が解析することにより生成した、例えば、図 6 に示したアセンブリコードが、ソースプログラム解析情報記憶部 1 2 に記憶されているものとして説明する。

指示情報受付部 1 8 は、操作者 3 1 から、例えば、キーボード装置やいわゆるマウスなどのポインティングデバイス、あるいは、コンパイルを補助するプログラムである例えばグラフィカルユーザインターフェース (GUI: Graphical User Interface) 3 2 を通じて入力された種々のコンパイル指示情報を受け付けて、これを実行形式生成部 1 3 に供給する。

ここで、コンパイル指示情報は、例えば、間接実行形式と、直接実行形式の別などの情報や、コンパイル処理を制御するための情報などである。コンパイル処理を制御するための情報としては、例えば、コードの配置を制御するための情報や、ある特定のブロックを最適化するように指示するなどの最適化を制御するための情報などがある。

また、指示情報受付部 1 8 は、目的とする実行形式プログラムを生成するためにコンパイル処理において必要となるパラメータや、コンパイル処理に必要となるデータが蓄積されたファイルの指示情

報などを与えるコンパイル処理を補助するためのプログラムからのコンパイル指示情報を受け付けて、これを実行形式生成部 13 に供給する。

そして、実行形式生成部 13 は、指示情報受付部 18 からのコンパイル指示情報をも考慮して、ソースプログラム解析情報記憶部 12 の解析情報から、又は、ソースプログラム解析情報記憶部 12 の解析情報と、プロファイル情報記憶部 15 のプロファイル情報とから、コンパイル指示情報に基づいた実行形式プログラムを生成する。

これにより、操作者やコンパイル処理を補助するためのプログラムからのコンパイル指示情報をも考慮して、最適化された目的とする実行形式プログラムを簡単かつ迅速に生成することができる。

次のこの第 3 の実施の形態のコンパイル処理装置 300 について、具体例を用いてより詳細に説明する。

この第 3 の実施の形態のコンパイル処理装置 300 においては、ソースプログラム解析部 11 における解析結果や、実行形式生成部 13 において生成された実行形式プログラムの生成結果などの情報、あるいは、プロファイル情報を統計処理することにより形成される確率情報などの各種の情報が、この実施の形態のコンパイル処理装置 300 に接続されたディスプレイに表示されるなどして操作者に通知することができるようにされている。

操作者は、ディスプレイに表示された情報を確認しながら各種のコンパイル指示情報を入力することができるようにされる。すなわち、この第 3 の実施の形態においては、グラフィカルユーザインターフェース 32 が実現され、このグラフィカルユーザインターフェースを通じてコンパイル指示情報をコンパイル処理装置 300 に

入力することができるようにされている。

そして、操作者は、例えばディスプレイに表示されるなどして通知された情報などに基づいて、キーボードやマウスなどのポインティングデバイスや、グラフィカルユーザインターフェースを通じて、コンパイル指示情報を入力する。入力されたコンパイル指示情報は、指示情報受付部 18 を通じて、実行形式生成部 13 に供給される。

図 13 は、この第 3 の実施の形態のコンパイル処理装置 300 の指示情報受付部 18 と、この指示情報受付部 18 からのコンパイル指示情報の供給を受ける実行形式生成部 13 とを説明するための図である。例えば、コンパイル指示情報が、間接実行形式と直接実行形式の別を指示する情報である場合には、受け付けられたコンパイル指示情報は、実行形式生成部 13 のプログラム構築部 131 に供給される。

また、受け付けられたコンパイル指示情報が、プロファイル情報から生成された確率情報を変更することによって、目的とする最適化を行うようにする最適化を制御するための情報である場合には、受け付けられた情報は、確率情報生成部 132 に供給される。

図 14 は、グラフィカルユーザインターフェース 32 の表示例を示す図である。この例は、前述したように、確率情報生成部 132 により、プロファイル情報から生成された確率情報（図 8）のうち、分岐の確率を示すとともに、分岐の確率の変更を行うための画面を示している。

前述もしたように、図 6 に示したアセンブリコードの場合を例にとると、ブロック 2 を実行する確率（変数 a が変数 b よりも大きい確率）は 10% であり、図 14 において $T = 10\%$ のように表示さ

れている。また、ブロック 3 が実行される確率（変数 b が変数 a 以上である確率）は 90% であり、図 14 において $F = 90\%$ のように表示されている。

この分岐の確率を変更して最適化を行いたい場合には、図 14 において矢印が示すように、最適化を制御するための情報の変更入力領域において、確率を変更する。図 14 の例の場合には、分岐の確率を逆にし、ブロック 2 を実行する確率（変数 a が変数 b よりも大きい確率）を 90% に変更し、ブロック 3 が実行される確率（変数 b が変数 a 以上である確率）を 10% に変更するようにしている。

この例の場合の最適化を制御するための情報は、図 13 を用いて前述したように、実行形式生成部 13 の確率情報生成部 132 に供給され、分岐の確率情報が変更するようにされる。そして、実行形式生成部 13 においては、指示情報受付部 18 から供給されたコンパイル処理を補助するためのプログラム、例えば、グラフィックルユーザインターフェースからのコンパイル指示情報をも考慮して、前述したように、実行形式プログラムを生成する。

図 15 は、図 14 を用いて前述したように、分岐の確率情報が変更されて最適化が行われて生成された実行形式プログラムをアセンブリコードで表したものである。図 6 に示したアセンブリコードが最適化前のアセンブリコードであり、分岐の確率が変わることにより、図 15 に示すように、block 1 → block 2 → block 3 の経路が最適化される。

この図 15 に示す最適化されたアセンブリコードと、図 6 に示した最適化される前のアセンブリコードとを比較すると分かるように、確率情報を用いて最適化を行うと、最適化後のアセンブリコードの

b l o c k 1 の 3 行 目 に は、新 た に レジスタ r 1 の 値 (変数 a) を レジスタ r 4 に 格納 する コード が 追 加 さ れ、b l o c k 2 の コード と、b l o c k 4 の コード が 統 合 する よう に さ れ て い る。

こ れ に よ り、b l o c k 1 に お い て、レジスタ r 1 の 値 (変数 a) が レジスタ r 4 に 格納 さ れ て い る の で、変数 a が 変数 b よ り 大 き い と き に は、b l o c k 1 に 続 く b l o c k 2 に お い て、変数 a の 2 乗 が 即 座 に 求 め ら れ 変数 d に 格納 する こ と が で き る よう に さ れ る。す な わ ち、2 つ の b l o c k の 処 理 で、変数 2 の 2 乗 を 求 め る こ と が で き る よう に さ れ る。

ま た、変数 b が 変数 a よ り 大 き い 場 合 に は、図 6 に 示 し た 最 適 化 前 の ア セ ン ブ リ コード の 場 合 と 同 様 に、b l o c k 3 に お い て 変数 b が 変数 c に 格納 さ れ、b l o c k 4 に お い て、変数 c (す な わ ち 変数 a) が 2 乗 さ れ て 変数 d に 格納 さ れ る と い う よう に、3 つ の ブ ロ ッ ク の 処 理 で、変数 a の 2 乗 を 求 め る こ と が で き る よう に さ れ る。

し た が っ て、発 生 する 確 率 が 9 0 % と さ れ た 変数 a が 変数 b よ り も 大 き い 場 合 の 処 理 が、迅 速 に 行 わ れ る よう に、最 適 化 さ れ た 実 行 形 式 プ ロ グ ラ ム が 生 成 さ れ る。な お、図 1 5 に お い て は 示 さ な か っ た が、b l o c k 2 か ら ジ ャ ンプ する 先 の b l o c k 5 は、b l o c k 4 に 続 く ブ ロ ッ ク で あ る。

こ の よう に、こ の 第 3 の 実 施 の 形 態 の コ ン パ イ ル 処 理 装 置 3 0 0 は、操 作者 や コ ン パ イ ル 処 理 を 補 助 する た め の プ ロ グ ラ ム か ら の コ ン パ イ ル 指 示 情 報 を も 考 慮 し て、最 適 化 さ れ た 目 的 と する 実 行 形 式 プ ロ グ ラ ム を 簡 単 か つ 迅 速 に 生 成 する こ と が で き る。つ ま り、操 作者 は、手 動 で コ ン パ イ ル 指 示 情 報 を コ ン パ イ ル 処 理 装 置 3 0 0 に 対 し て 与 え る こ と が で き る の で、コ ン パ イ ル 処 理 装 置 が 自 動 で は で き

ない細かい制御を行うことが可能になり、より効率のよい実行形式プログラムを生成することができる。

また、例えば、操作者は、プロファイル情報を得るためにソースプログラム解析情報記憶部 12 に記憶されている解析情報に基づいた実行形式プログラムを作成するのか、プロファイル情報を考慮して、最適化した実行形式プログラムを生成するのかなどの指示も、指示情報受付部 18 を通じて、実行形式生成部 13 に供給することができる。

[第 4 の実施の形態]

前述した第 3 の実施の形態においては、コンパイル処理の実行時にコンパイル指示情報を指示情報受付部 18 を通じて、実行形式生成部 13 に供給するようにした。しかし、通常コンパイル指示情報は、そのほとんどがコンパイル処理前には分かっている場合が多い。

このため、コンパイル処理前に分かっているコンパイル指示情報については、予めコンパイル処理装置に与えておき、コンパイル処理装置においてのコンパイル処理を迅速に行うようにするとともに、例えば、コンパイル指示情報が間違っていた場合などにおいては、間違っていたコンパイル指示情報を修整するだけで、すなわち、再度、すべてのコンパイル指示情報の入力をすることなくコンパイル処理を行って、実行形式プログラムを迅速に生成することができるようにしておくことが望ましい。

そこで、この第 4 の実施の形態のコンパイル処理装置は、操作者やコンパイル処理を補助するためのプログラムからのコンパイル指示情報を記憶保持するコンパイル指示情報記憶部を設け、コンパイル処理を実行する都度、コンパイル指示情報の入力をするこ

いようにしたものである。

図 1 6 は、この第 4 の実施の形態のコンパイル処理装置 4 0 0 を説明するための図である。図 1 6 に示すように、この第 4 の実施の形態のコンパイル処理装置 4 0 0 は、コンパイル指示情報記憶部 1 9 以外の各部は、図 1 2 を用いて前述した第 3 の実施の形態のコンパイル処理装置 3 0 0 の各部と同様に構成されたものであり、指示情報受付部 1 8、コンパイル指示情報記憶部 1 9 を除けば、第 1 の実施の形態のコンパイル処理装置 1 0 0 と同様に構成されたものである。

そして、この第 4 の実施の形態のコンパイル処理装置 4 0 0 において、指示情報受付手段 1 8 は、操作者やコンパイル処理を補助するためのプログラムからのコンパイル指示情報を受け付けて、これをコンパイル指示情報記憶部 1 9 に記憶する。

コンパイル指示情報記憶部 1 9 に記憶されたコンパイル指示情報は、実行形式プログラムの生成時に、実行形式生成部 1 3 により読み出される。そして、実行形式生成部 1 3 は、コンパイル指示情報記憶部 1 9 から読み出したコンパイル指示情報をも考慮して、実行形式プログラムを生成する。

これにより、操作者やコンパイル処理を補助するためのプログラムからのコンパイル指示情報を予めコンパイル指示情報記憶部 1 9 に記憶させておくことにより、コンパイル処理実行時において、予め決まっていた数多くのコンパイル指示情報を入力したり、コンパイル処理を補助するためのプログラムを実行しなくても済むので、より迅速に、目的とする計算機で実行可能とされた目的の実行形式の実行形式プログラムを生成することができる。

つまり、コンパイル処理時においてコンパイル処理装置に提供するコンパイル指示情報を必要最小限に抑さえることができるなど、コンパイル指示情報の入力にかかる時間を短縮することができ、目的とする実行形式プログラムを迅速に生成することができる。

なお、前述の実施の形態において示したソースプログラムや中間コード、アセンブリコード、プロファイル情報、確率情報などは一例であり、この発明のコンパイル処理装置を用いることによって、各種のソースプログラムからそのソースプログラムに応じた解析情報の生成、実行形式プログラムの生成、プロファイル情報の生成、最適化した実行形式プログラムの生成を行うことができる。

また、コンパイル指示情報は、前述した実行形式の別や、目的とする計算機のプログラムの実行環境に関する情報、コンパイルを制御する情報に限るものではなく、目的の実行形式プログラムを生成するためにコンパイル処理において必要となる種々の情報をも含むものである。

また、前述の第3、第4の実施の形態のコンパイル処理装置300、400に、前述した第2の実施の形態の間接実行形式中間生成部17を搭載することもできる。この場合には、操作者などからの中間コードプログラムを生成する指示を間接実行形式中間生成部17に供給して、中間コードプログラムを生成するようにすることができる。

また、前述した実施の形態のコンパイル処理装置は、ソフトウェアによって構成するようにすることができる。すなわち、前述した各実施の形態のコンパイル処理装置の機能を備えた、コンパイル処理プログラムを形成することができる。

また、前述した実施の形態のコンパイル処理装置の実行形式生成部は、関連する複数の実行形式プログラムを相互にリンクさせて、1つの実行形式モジュールを生成するようにすることもできるものである。

また、前述した各実施の形態のコンパイル処理装置は、単独で使用するようにすることもできるが、種々の電子計算機に搭載して利用するようにすることができる。

また、前述した各実施の形態のコンパイル処理装置は、扱うプログラムのデータ構造などに依存することなく、目的とする計算機において実行可能な実行形式プログラムを生成することができる。

すなわち、前述した各実施の形態のコンパイル処理装置は、これが搭載される計算機のハードウェア構成や、扱うプログラムなどのデータのデータ構造などに左右されることなく、目的とする計算機などの電子機器において実行可能な実行形式プログラムを生成することができるものである。

以上説明したように、本発明に係るコンパイル処理装置によれば、ソースプログラムから、そのソースプログラムの解析情報を生成し、この解析情報に基づいて第1の実行形式プログラムを生成し、この第1の実行形式プログラムに基づいてプロファイル情報を生成し、前記前記解析情報と前記プロファイル情報とに基づいて第2の実行形式プログラムを生成するので、最適化された実行形式プログラムを生成する場合に、ソースプログラムを繰り返し用いることなく、ソースプログラムを解析処理することにより生成されたソースプログラムの解析情報とプロファイル情報とに基づいて、最適化された実行形式のプログラムを生成することができる。したがって、最適

化された実行形式のプログラムを生成する場合に、プロファイル情報を生成するための実行形式のプログラムを生成するために行ったソースプログラムについての解析処理を重複して行う必要がなくなり、迅速に最適化した実行形式プログラムを生成することができる。

また、本発明に係るコンパイル処理装置によれば、前記解析情報を記憶しておくことにより、プロファイル情報を得るために実行するプロファイル情報生成用の最適化前の実行形式プログラムも、最適化された実行形式プログラムも、同じ実行形式プログラム生成部により生成することができる。したがって、プロファイル情報生成用実行形式プログラム生成部と、最適化実行形式プログラム生成部とを別々に設ける必要がなくなり、コンパイル処理装置の構成を簡単にすることができる。

また、本発明に係るコンパイル処理装置によれば、前記プロファイル情報を記憶しておくことにより、目的とする計算機において実行可能であるその計算機に固有な形式であって、その目的とする計算機のプログラムの実行環境などに適合したプログラムを生成することができる。また、この直接実行形式のプログラムをコンパイル処理装置を有する装置において実行することにより、その直接実行形式のプログラムのプロファイル情報を得て、最適化された実行形式のプログラムを生成することができる。

また、本発明に係るコンパイル処理装置によれば、間接実行形式のプログラムが生成できる。間接実行形式のプログラムは、これを実行する計算機などの機器に依存することなく実行することができるので、コンパイル処理装置において、間接実行形式のプログラムを実行することにより、プロファイル情報を得て、このプロファイ

ル情報をも用いることによって、最適化した実行形式のプログラムを作成することができる。

また、本発明に係るコンパイル処理装置によれば、プログラムが実行される計算機のプログラムの実行環境などに依存することなく、間接実行形式中間プログラム生成部により生成されるいわゆる中間コードプログラムを実行することにより、そのプログラムに対するプロファイル情報を得て、このプロファイル情報を用いることにより、目的とする間接実行形式の最適化されたプログラムを簡単かつ迅速に生成することができる。

また、本発明に係るコンパイル処理装置によれば、操作者からのコンパイル指示情報をも考慮した実行形式プログラムを簡単かつ迅速に生成することができる。

また、本発明に係るコンパイル処理装置によれば、操作者は、コンパイル処理前に予め決まっているコンパイル指示情報について、予めコンパイル指示情報記憶部に記憶しておくことができる。したがって、コンパイル処理実行時のコンパイル指示情報の入力処理にかかる時間を短縮することができ、コンパイル指示情報記憶部に記憶されたコンパイル指示情報をも考慮した、操作者からの要求に応じた実行形式プログラムをより迅速に生成することができる。

また、本発明に係るコンパイル処理装置によれば、コンパイル処理を補助するためのプログラムからのコンパイル指示情報をも考慮して、実行形式プログラムが生成されるので、目的とする実行形式プログラムを簡単かつ迅速に生成することができる。

さらに、本発明に係るコンパイル処理装置によれば、コンパイル処理に必要なコンパイル処理を補助するプログラムからの種々のコ

ンパイル指示情報を予めコンパイル指示情報記憶部に記憶しておくことができるので、コンパイル処理時にコンパイル処理を補助するためのプログラムを実行させる必要がなくなる。これにより、コンパイル指示情報記憶部に記憶されたコンパイル指示情報をも考慮して、目的とする実行形式プログラムをより迅速に生成することができる。

請 求 の 範 囲

1. ソースプログラムから、そのソースプログラムの解析情報を生成する解析情報生成部と、

前記解析情報に基づいて、第1の実行形式プログラムを生成する第1の実行形式プログラム生成部と、

前記第1の実行形式プログラムに基づいて、プロファイル情報を生成するプロファイル情報生成部と、

前記前記解析情報と、前記プロファイル情報とに基づいて、第2の実行形式プログラムを生成する第2の実行形式プログラム生成部と

を備えることを特徴とするコンパイル処理装置。

2. 前記解析情報生成部により生成される前記解析情報を記憶する解析情報記憶部を備えることを特徴とする請求の範囲第1項に記載のコンパイル処理装置。

3. 前記プロファイル情報生成部により生成される前記プロファイル情報を記憶するプロファイル情報記憶部を備えることを特徴とする請求の範囲第1項に記載のコンパイル処理装置。

4. 前記第1の実行形式プログラム生成部と、前記第2の実行形式プログラム生成部とは、1つの実行形式プログラム生成部により構成されることを特徴とする請求の範囲第1項に記載のコンパイル処理装置。

5. 前記第1の実行形式プログラム生成部は、目的とする計算機において実行される直接実行形式のプログラムを生成することを特

徴とする請求の範囲第 1 項に記載のコンパイル処理装置。

6. 前記第 1 の実行形式プログラム生成部は、間接実行形式のプログラムを生成することを特徴とする請求の範囲第 1 項に記載のコンパイル処理装置。

7. 前記第 1 の実行形式プログラム生成部は、プログラムを実行する計算機に依存することなく実行可能な間接実行形式の中間プログラムを生成する間接実行形式中間プログラム生成部であることを特徴とする請求の範囲第 6 項に記載のコンパイル処理装置。

8. 操作者からのコンパイル指示情報の入力を受け付けるコンパイル指示情報受付部を備え、

前記第 2 の実行形式プログラム生成部は、前記コンパイル指示情報受付部を通じて入力された前記コンパイル指示情報と、前記解析情報と、前記プロファイル情報とに基づいて、目的とする計算機の最適化された実行形式のプログラムを生成することを特徴とする請求の範囲第 1 項に記載のコンパイル処理装置。

9. 操作者からのコンパイル指示情報の入力を受け付けるコンパイル指示情報受付部と、

前記コンパイル指示情報受付部を通じて入力された前記コンパイル指示情報を記憶する操作者用のコンパイル指示情報記憶部とを備え、

前記第 2 の実行形式プログラム生成部は、前記コンパイル指示情報記憶部の前記コンパイル指示情報と、前記解析情報と、前記プロファイル情報とに基づいて、目的とする計算機の最適化された実行形式のプログラムを生成することを特徴とする請求の範囲第 1 項に記載のコンパイル処理装置。

10. コンパイル処理を補助するためのプログラムからコンパイル指示情報を受け付けるコンパイル指示情報受付部を備え、

前記第2の実行形式プログラム生成部は、前記コンパイル指示情報受付部を通じて入力された前記コンパイル指示情報と、前記解析情報と、前記プロファイル情報とに基づいて、目的とする計算機の最適化された実行形式のプログラムを生成することを特徴とする請求の範囲第1項に記載のコンパイル処理装置。

11. コンパイル処理を補助するためのプログラムからのコンパイル指示情報を受け付けるコンパイル指示情報受付部と、

前記コンパイル指示情報受付部を通じて受け付けた前記コンパイル指示情報を記憶するコンパイル指示情報記憶部とを備え、

前記第2の実行形式プログラム生成部は、前記コンパイル指示情報記憶部の前記コンパイル指示情報と、前記解析情報と、前記プロファイル情報とに基づいて、目的とする計算機の最適化された実行形式のプログラムを生成することを特徴とする請求の範囲第1項に記載のコンパイル処理装置。

12. ソースプログラムから、そのソースプログラムの解析情報を生成する解析情報生成工程と、

前記解析情報に基づいて、第1の実行形式プログラムを生成する第1の実行形式プログラム生成工程と、

前記第1の実行形式プログラムに基づいて、プロファイル情報を生成するプロファイル情報生成工程と、

前記解析情報と、前記プロファイル情報とに基づいて、第2の実行形式プログラムを生成する第2の実行形式プログラム生成工程と

を有するコンパイル処理方法。

13. 前記解析情報生成工程により生成される前記解析情報を、解析情報記憶部に記録する解析情報記録工程を有することを特徴とする請求の範囲第12項に記載のコンパイル処理方法。

14. 前記プロファイル情報生成工程により生成される前記プロファイル情報をプロファイル情報記憶部に記録するプロファイル情報記録工程を有することを特徴とする請求の範囲第12項に記載のコンパイル処理方法。

15. 前記第1の実行形式プログラム生成工程は、目的とする計算機において実行される直接実行形式のプログラムを生成することを特徴とする請求の範囲第12項に記載のコンパイル処理方法。

16. 前記第1の実行形式プログラム生成工程は、間接実行形式のプログラムを生成することを特徴とする請求の範囲第12項に記載のコンパイル処理方法。

17. 前記第1の実行形式プログラム生成工程は、プログラムを実行する計算機に依存することなく実行可能な間接実行形式の中間プログラムを生成する間接実行形式中間プログラム生成工程であることを特徴とする請求の範囲第16項に記載のコンパイル処理方法。

18. 操作者からのコンパイル指示情報の入力を受け付けるコンパイル指示情報受付工程を備え、

前記第2の実行形式プログラム生成工程においては、前記コンパイル指示情報受付工程において受け付けられた前記コンパイル指示情報と、前記解析情報と、前記プロファイル情報とに基づいて、目的とする計算機の最適化された実行形式のプログラムを生成することを特徴とする請求の範囲第12項に記載のコンパイル処理方法。

19. 操作者からのコンパイル指示情報の入力を受け付けて、これをコンパイル指示情報記憶部に記憶するコンパイル指示情報受付工程を備え、

前記第2の実行形式プログラム生成工程においては、前記コンパイル指示情報記憶部の前記コンパイル指示情報と、前記解析情報と、前記プロファイル情報とに基づいて、目的とする計算機の最適化された実行形式のプログラムを生成することを特徴とする請求の範囲第12項に記載のコンパイル処理方法。

20. コンパイル処理を補助するためのプログラムからコンパイル指示情報を受け付けるコンパイル指示情報受付工程を備え、

前記第2の実行形式プログラム生成工程においては、前記コンパイル指示情報受付工程において受け付けられた前記コンパイル指示情報と、前記解析情報と、前記プロファイル情報とに基づいて、目的とする計算機の最適化された実行形式のプログラムを生成することを特徴とする請求の範囲第12項に記載のコンパイル処理方法。

21. コンパイル処理を補助するためのプログラムからのコンパイル指示情報を受け付けて、これをコンパイル指示情報記憶部に記憶するコンパイル指示情報受付工程を備え、

前記第2の実行形式プログラム生成工程においては、前記コンパイル指示情報記憶部の前記コンパイル指示情報と、前記解析情報と、前記プロファイル情報とに基づいて、目的とする計算機の実行形式のプログラムを生成することを特徴とする請求の範囲第12項に記載のコンパイル処理方法。

THIS PAGE BLANK (USPTO)

1/12

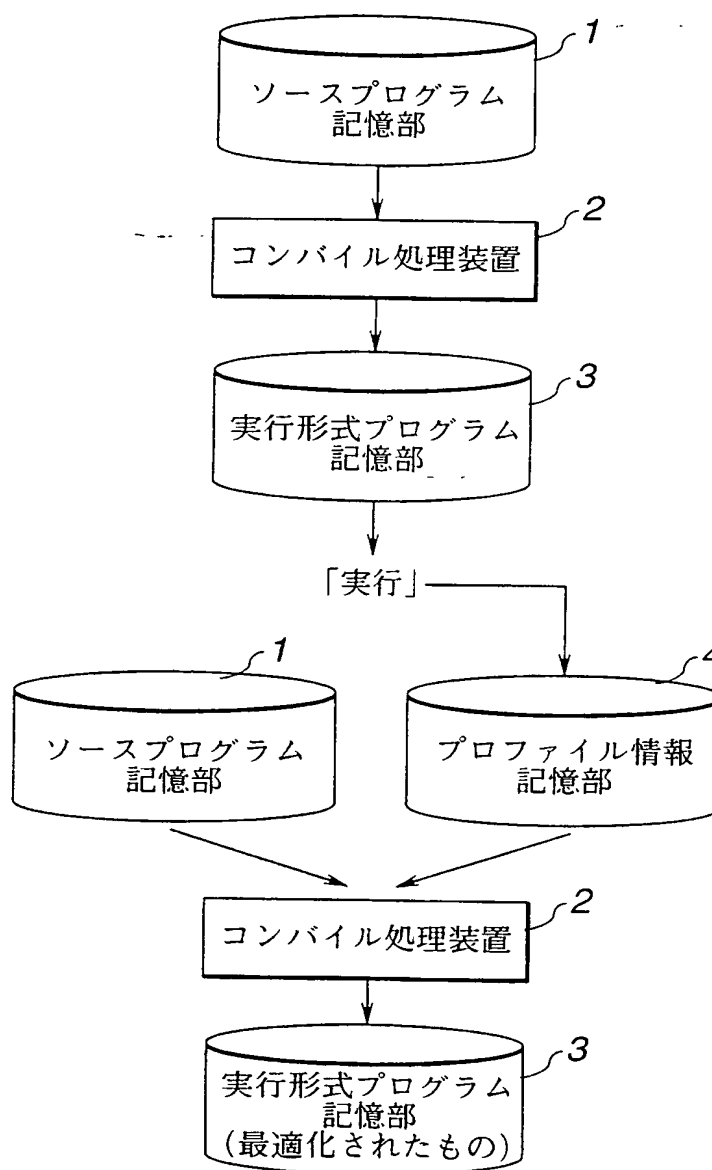


FIG.1

THIS PAGE BLANK (USPTO)

2/12

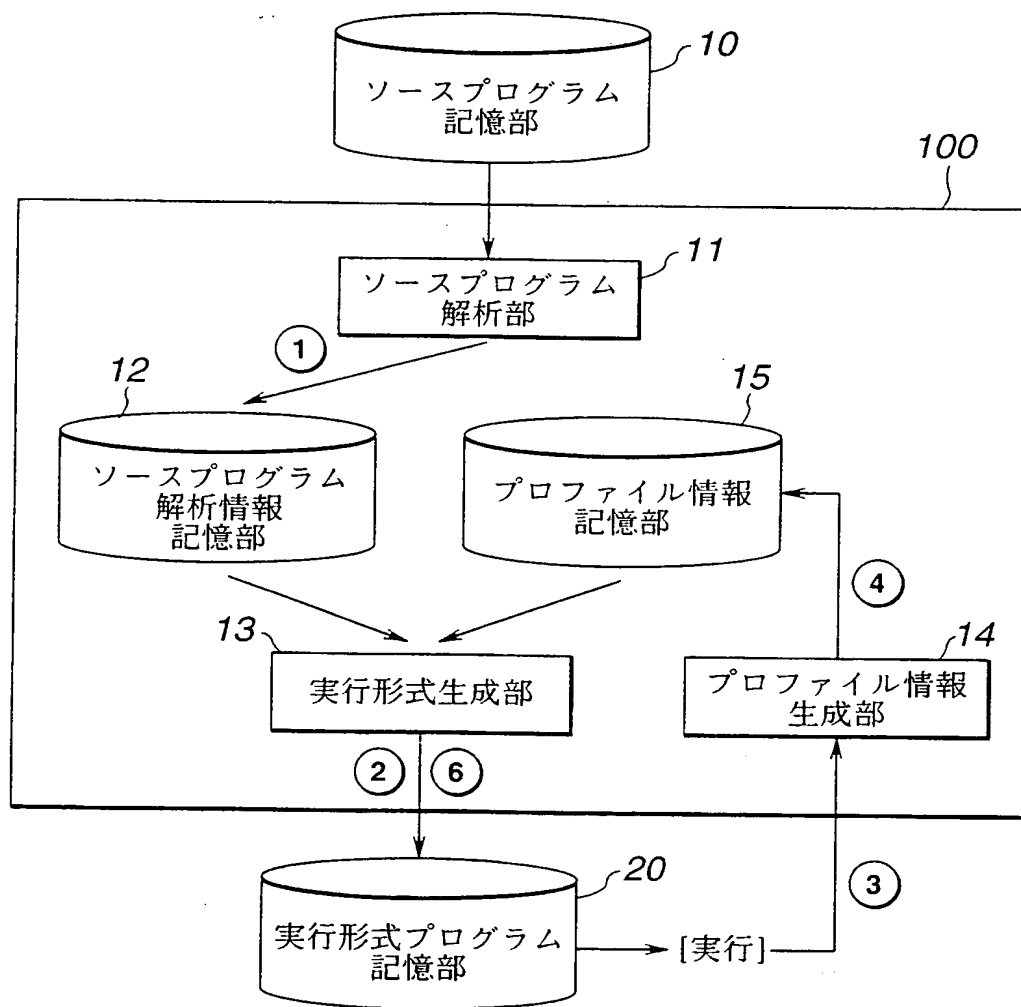


FIG.2

THIS PAGE BLANK (USPTO)

3/12

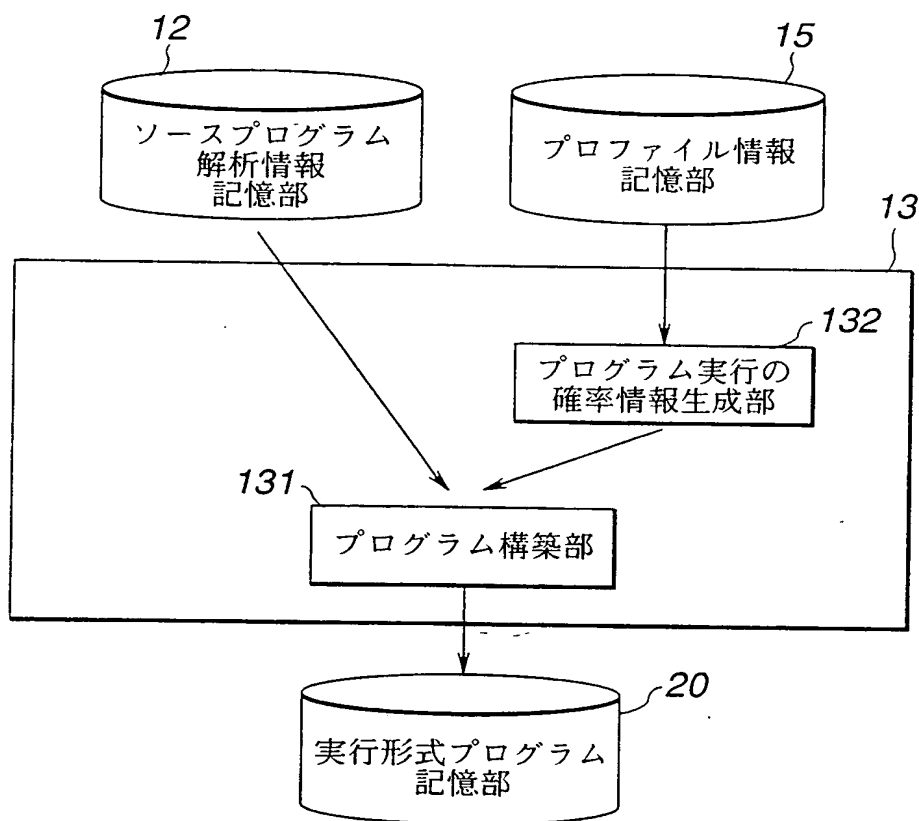


FIG.3

THIS PAGE BLANK (USPTO)

4/12

ソースプログラムの例

```
if (a > b)      then c = a ;  
                else c = b ;  
  
d = c * c ;
```

FIG.4

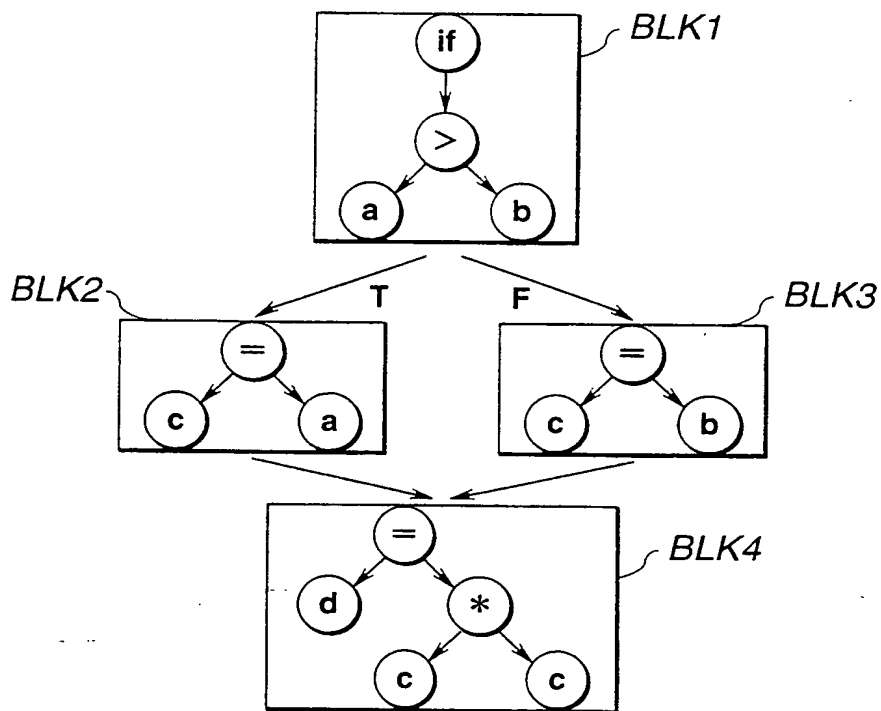


FIG.5

THIS PAGE BLANK (CSF10)

5/12

アセンブリコード

```

.text
block_1 :
    ld    r1, [a]        ; r1<-a
    ld    r2, [b]        ; r2<-b
    cmp   r3, r1, r2     ; r3<-r1 cmp r2
    ble   r3, block_3
block_2 :
    ld    r4, [a]        ; r4<-a
    st    r4, [c]        ; c<-r4
    jmp   block_4
block_3 :
    ld    r5, [b]        ; r5<-b
    st    r5, [c]        ; c<-r5
    jmp   block_4
block_4 :
    ld    r6, [c]        ; r6<-c
    ld    r7, [c]        ; r7<-c
    mul   r8, r6, r7     ; r8<-r6 * r7
    st    r8, [d]        ; d<-r8

```

FIG.6

プロファイル情報

経過時間

詳細情報

...		
10050:	PC	block_1
	load address	a
10051:	load address	b
10052:	compare	
10053:	branch less or equal	block_3
10054:	PC	block_3
	load address	b
10055:	store address	c
10056:	jump	block_4
10057:	PC	block_4
	load address	c
10058:	load address	c
10059:	mul	
10080:	store address	d
...		

FIG.7

THIS PAGE BLANK (USPTO)

6/12

確率情報

ラベル名

実行／アクセス回数

```

...
block_1 :      100
block_2 :      10
block_3 :      90
block_4 :      100
...
a load:        110
a store:        0
b load:        190
b store:        0
c load:        200
c store:        100
d load:         0
d store:        100
...

```

FIG.8

```

.text
block_1 :
    ld    r1, [a]        ; r1<-a
    ld    r2, [b]        ; r2<-b
    mv     r5, r2         ; r5<-r2
    cmp    r3, r1, r2     ; r3<-r1 cmp r2
    ble    r3, block_3
block_2 :
    ld    r4, [a]        ; r4<-a
    st     r4, [c]        ; c<-r4
    jmp    block_4
block_3 :
    st     r5, [c]        ; c<-r5
    mul    r8, r5, r5     ; r8<-r5 * r5
    st     r8, [d]        ; d<-r8
    jmp    block_5
block_4 :
    ld     r6, [c]        ; r6<-c
    ld     r7, [c]        ; r7<-c
    mul    r8, r6, r7     ; r8<-r6 * r7
    st     r8, [d]        ; d<-r8

```

FIG.9

THIS PAGE BLANK (USPTO)

7/12

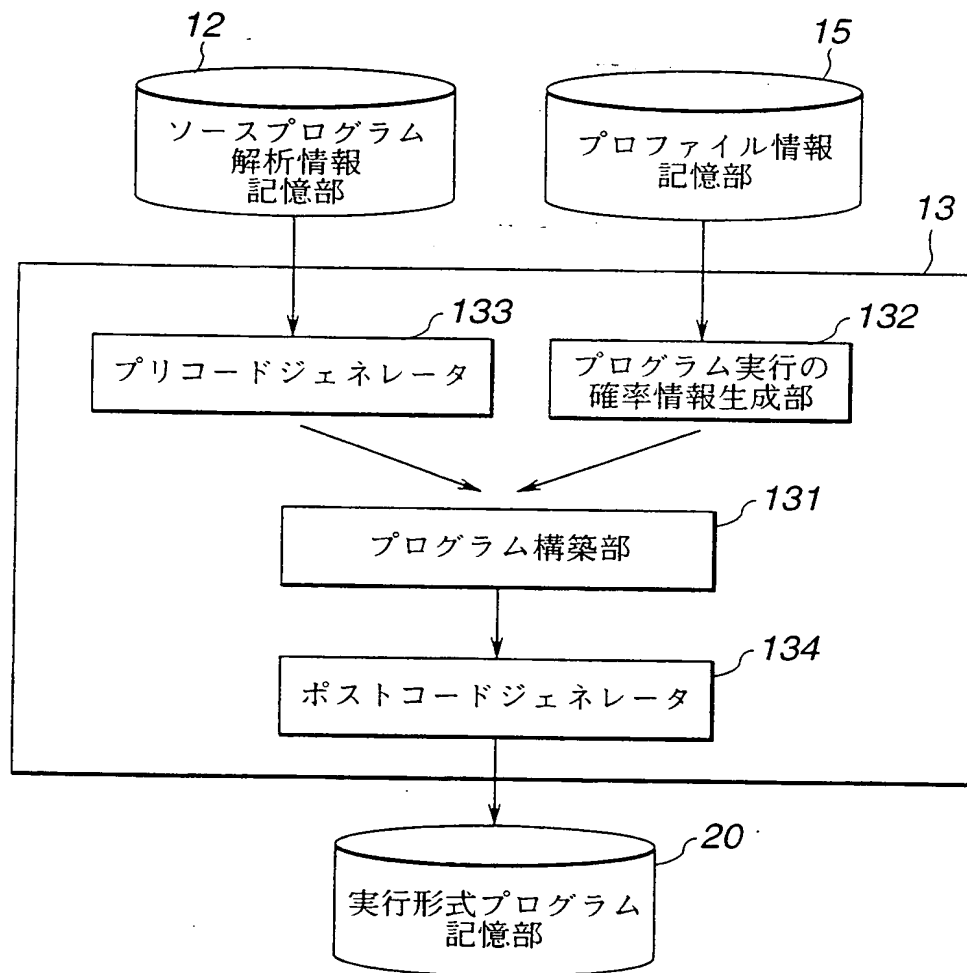


FIG.10

THIS PAGE BLANK (USPTO)

8/12

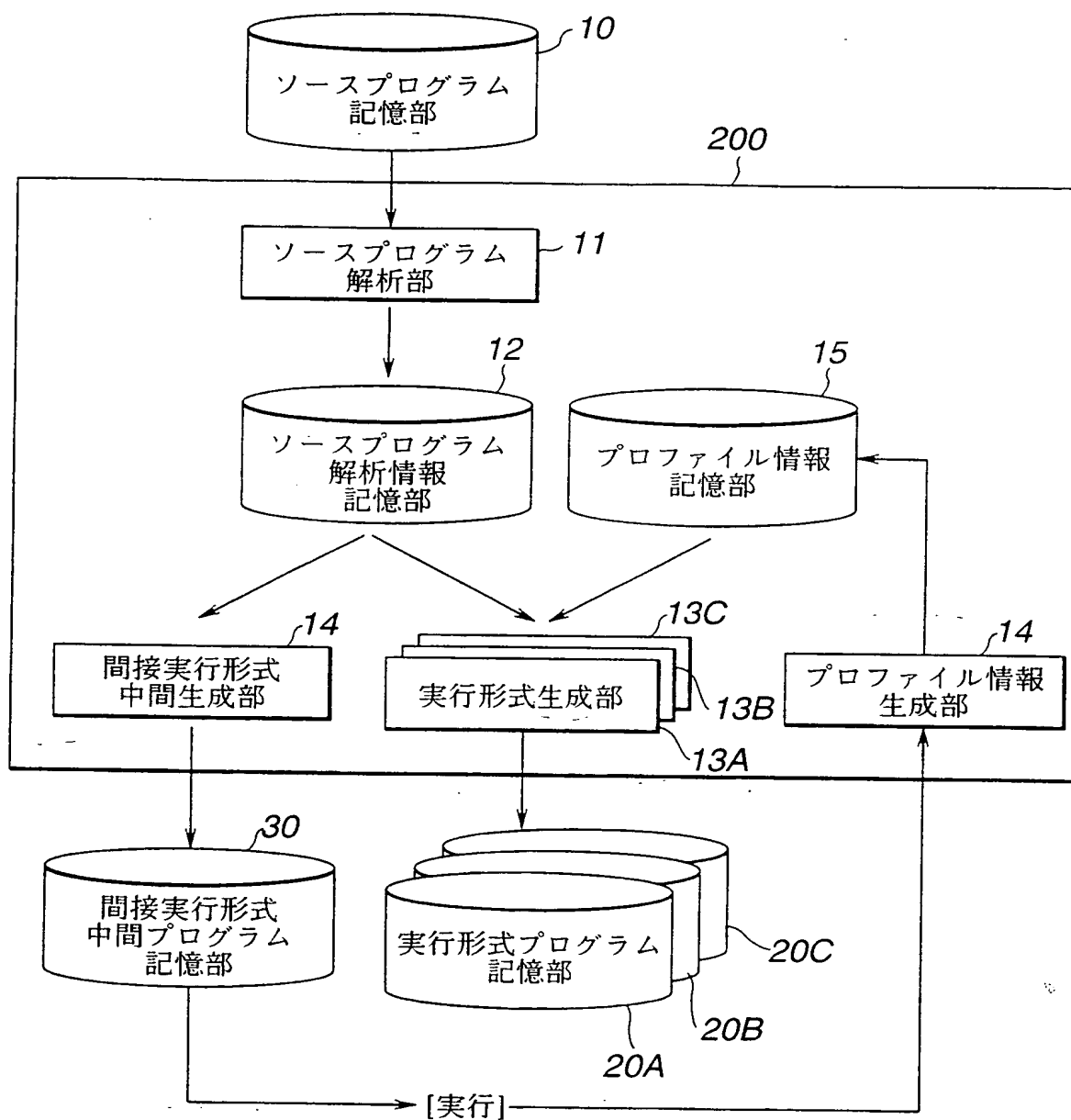


FIG.11

THIS PAGE BLANK (USPTO)

9/12

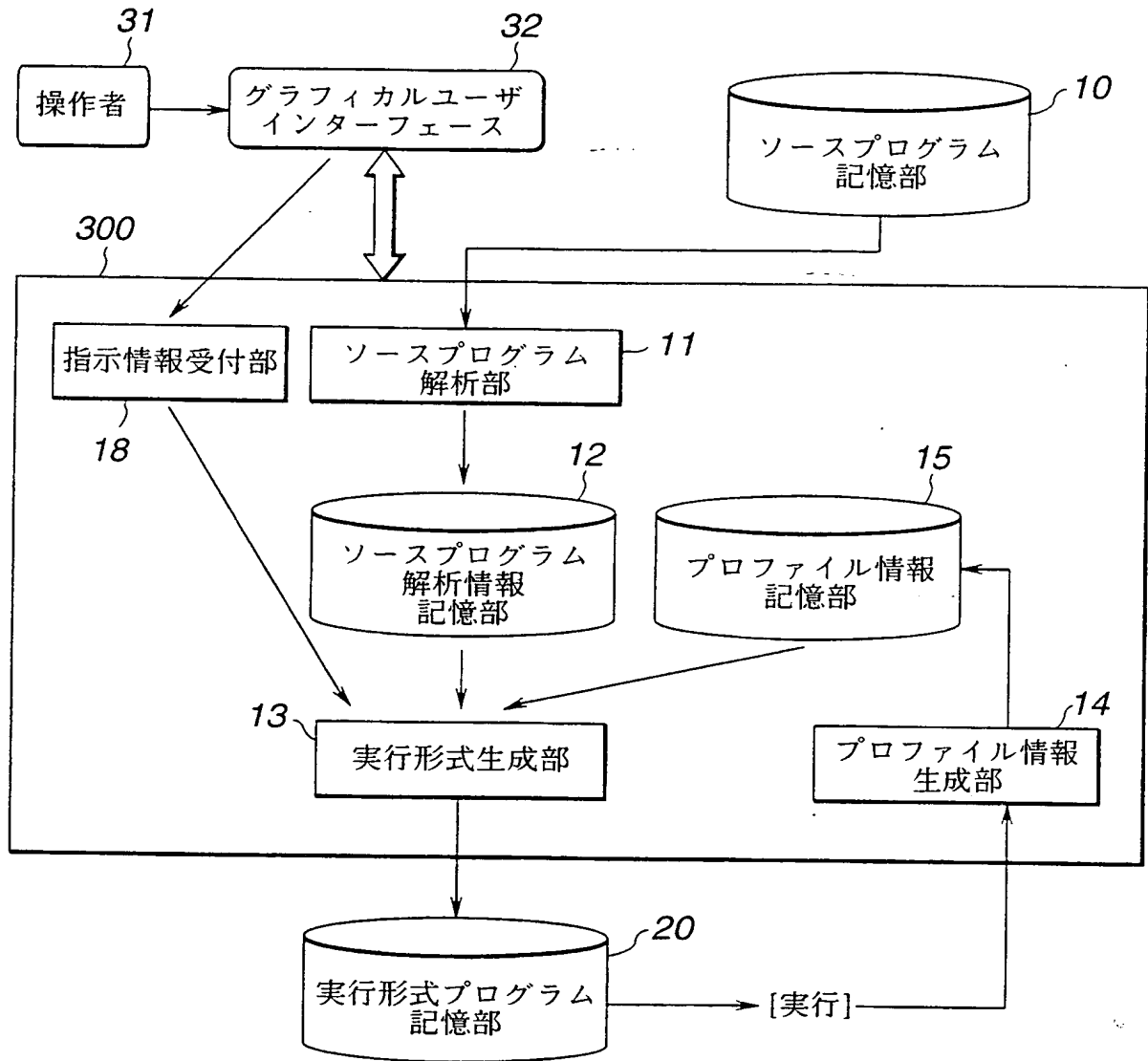


FIG.12

THIS PAGE BLANK (USPTO)

10/12

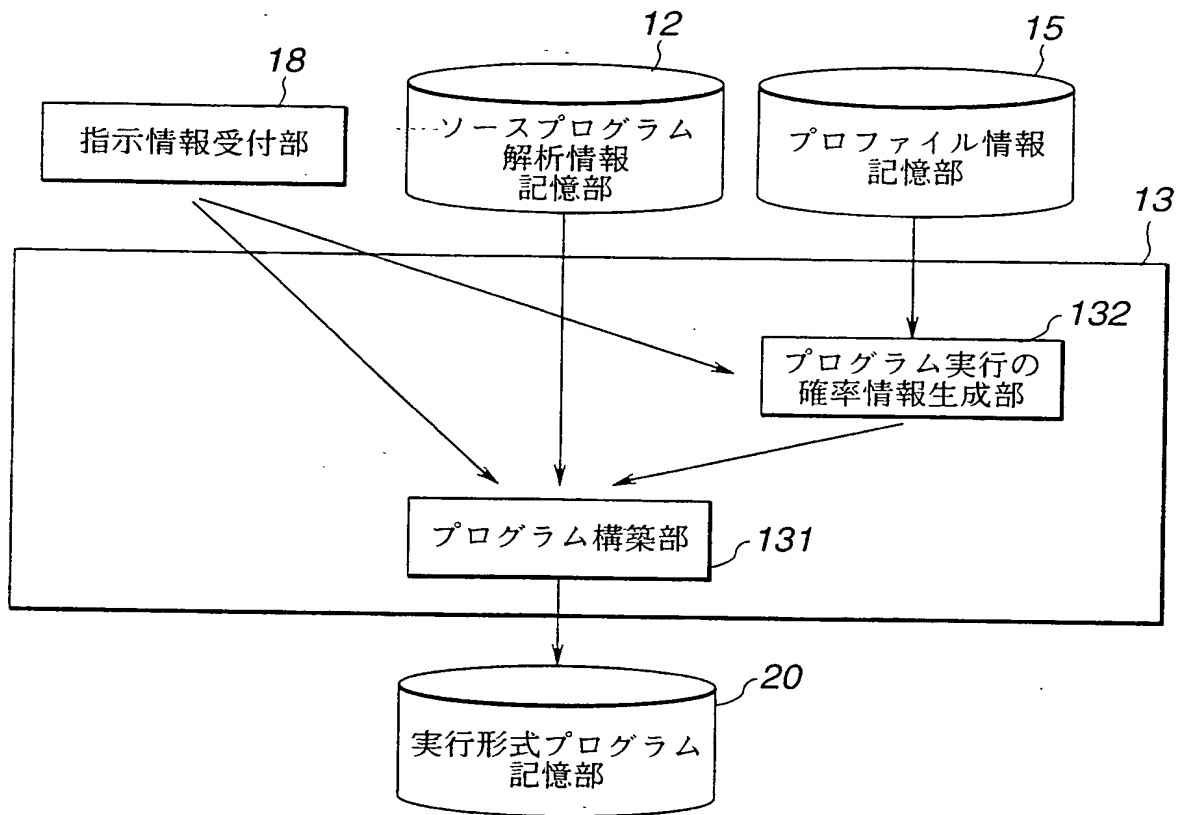


FIG.13

THIS PAGE BLANK (USPTO)

11/12

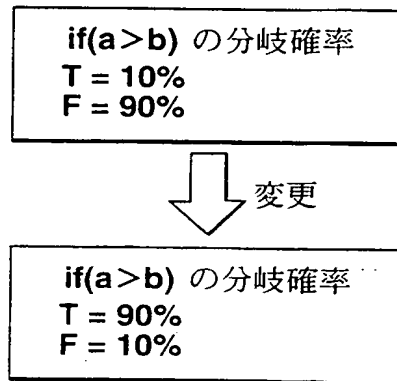


FIG.14

```

.text
block_1 :
    ld    r1, [a]           ; r1 <- a
    ld    r2, [b]           ; r2 <- b
    mv    r4, r1             ; r4 <- r1
    cmp   r3, r1, r2        ; r3 <- r1 cmp r2
    ble   r3, block_3
block_2 :
    ld    r4, [a]           ; c <- r4
    mul   r8, r4, r4         ; r8 <- r4 * r4
    st    r8, [d]           ; d <- r8
    jmp   block_5
block_3 :
    st    r5, [b]           ; r5 <- b
    st    r5, [c]           ; c <- r5
    jmp   block_4
block_4 :
    ld    r6, [c]           ; r6 <- c
    ld    r7, [c]           ; r7 <- c
    mul   r8, r6, r7        ; r8 <- r6 * r7
    st    r8, [d]           ; d <- r8

```

FIG.15

THIS PAGE BLANK (USPTO)

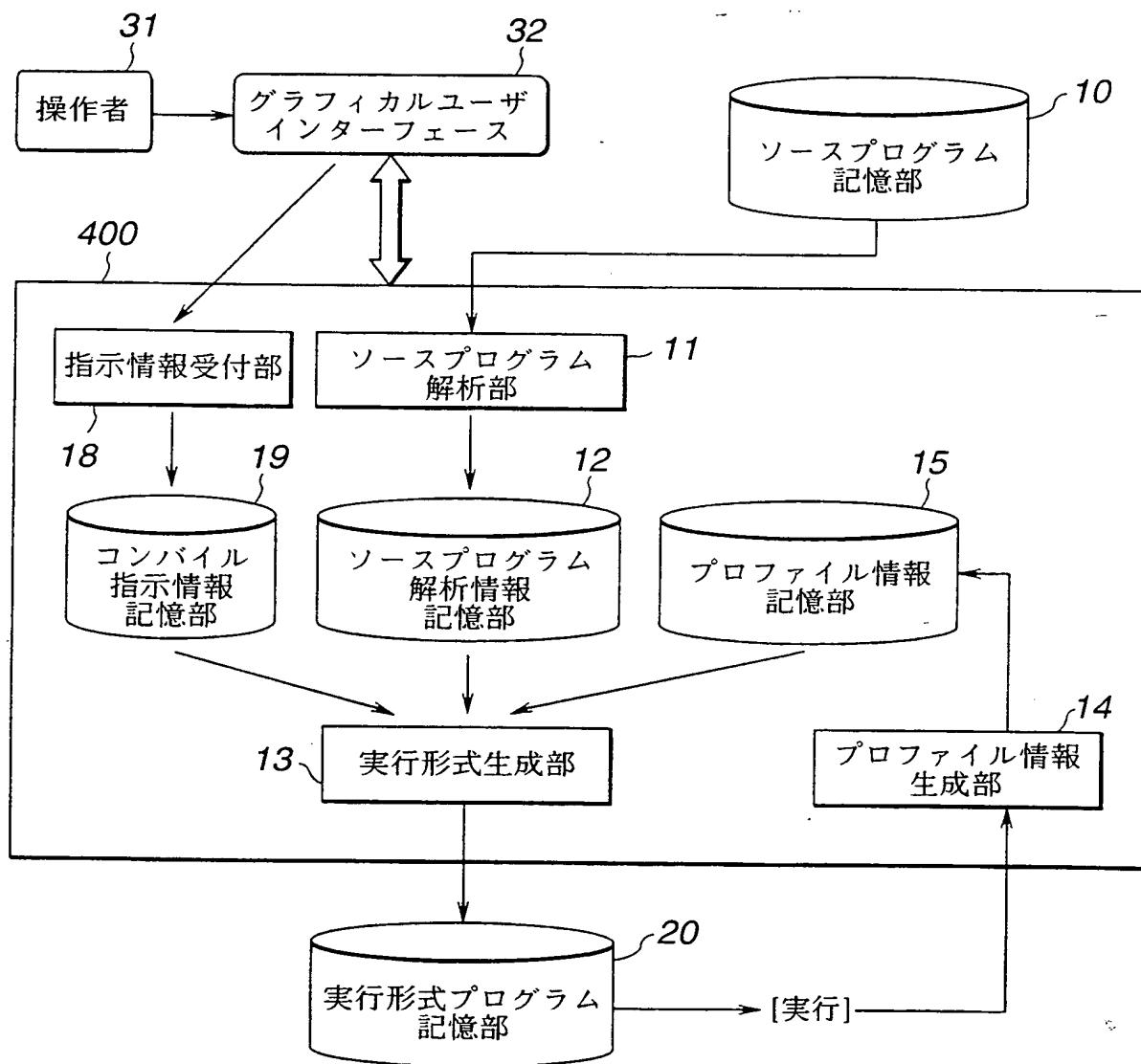


FIG.16

THIS PAGE BLANK (USPTO)

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP99/03710

A. CLASSIFICATION OF SUBJECT MATTER

Int.Cl⁶ G06F9/45

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
Int.Cl⁶ G06F9/45

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
Jitsuyo Shinan Koho 1971-1999 Toroku Jitsuyo Shinan Koho 1994-1999
Kokai Jitsuyo Shinan Koho 1971-1994

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
JICST File (JOIS)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US, 5659752, A (International Business Machines Co.), 19 August, 1997 (19. 08. 97)	1-21
Y	JP, 6-4299, A (NEC Software, Ltd.), 14 January, 1994 (14. 01. 94) (Family: none)	1-21
Y	JP, 63-276127, A (Fujitsu Ltd.), 14 November, 1988 (14. 11. 88), Fig. 2 ; page 3, upper right column, line 14 to lower right column, line 16 (particularly page 3, lower right column, lines 13 to 16) (Family: none)	1-21
Y	JP, 9-288580, A (NEC Corp.), 4 November, 1997 (04. 11. 97), Fig. 1 ; Par. Nos. [0043], [0065] (Family: none)	4
Y	JP, 9-330233, A (Hewlett-Packard Co.), 22 December, 1997 (22. 12. 97) & US, 5815720, A	6, 7, 16, 17

☒ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

* Special categories of cited documents:	"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"A" document defining the general state of the art which is not considered to be of particular relevance	"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"E" earlier document but published on or after the international filing date	"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	"&" document member of the same patent family
"O" document referring to an oral disclosure, use, exhibition or other means	
"P" document published prior to the international filing date but later than the priority date claimed	

Date of the actual completion of the international search
4 October, 1999 (04. 10. 99)

Date of mailing of the international search report
19 October, 1999 (19. 10. 99)

Name and mailing address of the ISA/
Japanese Patent Office

Authorized officer

Facsimile No.

Telephone No.

INTERNATIONAL SEARCH REPORT

International application No.

PCT/JP99/03710

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	JP, 6-202875, A (NEC Corp.), 22 July, 1994 (22. 07. 94), Fig. 4 ; Par. Nos. [0042] to [0050] (Family: none)	8-10, 18-20
Y	JP, 63-53646, A (NEC Corp.), 7 March, 1988 (07. 03. 88), Page 1, right column, lines 3 to 7 (Family: none)	11, 21

A. 発明の属する分野の分類 (国際特許分類 (IPC))
Int.Cl.⁸ G06F9/45

B. 調査を行った分野

調査を行った最小限資料 (国際特許分類 (IPC))
Int.Cl.⁸ G06F9/45

最小限資料以外の資料で調査を行った分野に含まれるもの

日本国実用新案公報 1971-1999年
日本国公開実用新案公報 1971-1994年
日本国登録実用新案公報 1994-1999年

国際調査で使用した電子データベース (データベースの名称、調査に使用した用語)
JICSTファイル (JOIS)

C. 関連すると認められる文献

引用文献の カテゴリー*	引用文献名 及び一部の箇所が関連するときは、その関連する箇所の表示	関連する 請求の範囲の番号
Y	US, 5 6 5 9 7 5 2, A (International Business Machines Co.) 19. 8月. 1997 (19. 08. 97)	1-21
Y	JP, 6-4299, A (日本電気ソフトウェア株式会社) 14. 1月. 1994 (14. 01. 94) (ファミリーなし)	1-21
Y	JP, 63-276127, A (富士通株式会社) 14. 11月. 1988 (14. 11. 88) 第2図、第3頁右上欄第14行~第3頁右下欄第16行 (特に、第3頁右下欄第13行~第16行) (ファミリーなし)	1-21

☒ C欄の続きにも文献が列挙されている。

☐ パテントファミリーに関する別紙を参照。

* 引用文献のカテゴリー

「A」 特に関連のある文献ではなく、一般的技術水準を示すもの

「E」 国際出願日前の出願または特許であるが、国際出願日以後に公表されたもの

「L」 優先権主張に疑義を提起する文献又は他の文献の発行日若しくは他の特別な理由を確立するために引用する文献 (理由を付す)

「O」 口頭による開示、使用、展示等に言及する文献

「P」 国際出願日前で、かつ優先権の主張の基礎となる出願

の日の後に公表された文献

「T」 国際出願日又は優先日後に公表された文献であって出願と矛盾するものではなく、発明の原理又は理論の理解のために引用するもの

「X」 特に関連のある文献であって、当該文献のみで発明の新規性又は進歩性がないと考えられるもの

「Y」 特に関連のある文献であって、当該文献と他の1以上の文献との、当業者にとって自明である組合せによって進歩性がないと考えられるもの

「&」 同一パテントファミリー文献

国際調査を完了した日

04. 10. 99

国際調査報告の発送日

19.10.99

国際調査機関の名称及びあて先

日本国特許庁 (ISA/JP)

郵便番号100-8915

東京都千代田区霞が関三丁目4番3号

特許庁審査官 (権限のある職員)

金田 利規

5B

9292

電話番号 03-3581-1101 内線 3545

C (続き) . 関連すると認められる文献		
引用文献の カテゴリー*	引用文献名 及び一部の箇所が関連するときは、その関連する箇所の表示	関連する 請求の範囲の番号
Y	JP, 9-288580, A (日本電気株式会社) 4. 11月. 1997 (04. 11. 97) 図1、段落0043, 0065 (ファミリーなし)	4
Y	JP, 9-330233, A (ヒューレット・パカード・カンパニー) 22. 12月. 1997 (22. 12. 97) & US, 5815720, A	6, 7, 16, 17
Y	JP, 6-202875, A (日本電気株式会社) 22. 7月. 1994 (22. 07. 94) 図4、段落0042~0050 (ファミリーなし)	8-10, 18-20
Y	JP, 63-53646, A (日本電気株式会社) 7. 3月. 1988 (07. 03. 88) 第1頁右欄第3行~第7行 (ファミリーなし)	11, 21